UNIVERSIDAD DE VALLADOLID

E.U. de Informática (Segovia)

Ingeniería Técnica en Informática de Gestión

SITEMA DE GESTIÓN Y CONTROL DE INVERNADERO AUTOMATIZADO CON ARDUINO

(HORTDUINO)

Alumno: Miguel Angel Hernanz Hernanz

Tutor: Fernando Díaz Gómez

BLOQUE I

MEMORIA DEL PFC

Índice de contenido

Índice de Ilustraciones

Índice de tablas

Índice de cuadros de código

1 INTRODUCCIÓN

1.1 IDENTIFICACIÓN DEL PROYECTO FIN DE CARRERA

Título: Sistema de gestión y control de invernadero automatizado con Arduino (*Hortduino*).

Autor: Miguel Angel Hernanz Hernanz

Director: Fernando Díaz Gómez

Departamento: Informática

Área: Ciencias de la Computación e Inteligencia Artificial

1.2 ORGANIZACIÓN DE LA DOCUMENTACIÓN

La documentación del proyecto Hortduino se estructura en tres secciones, siguiendo la recomendación de la guía de Proyectos Fin de Carrera. La primera sección se corresponde con la memoria del proyecto, la segunda se centra en la documentación técnica y la tercera contiene el manual de usuario.

La primera sección de esta memoria incluye la introducción e identificación del Proyecto Fin de Carrera (PFC) y otros elementos como:

- La descripción general del PFC, incluyendo los objetivos propuestos y cuestiones metodológicas relativas al mismo, así como el marco tecnológico en el que se desarrolla.
- La descripción del software implementado en el PFC destacando las funcionalidades básicas del producto y la interacción con usuarios y otros sistemas. También se detalla la arquitectura y los métodos empleados en el desarrollo del proyecto.
- Planificación y presupuesto del proyecto mediante el uso del análisis por puntos de función y modelo constructivo de costes COCOMO (Constructive Cost Model)
- Cuestiones de implementación y diseño a destacar en el desarrollo del software y del prototipado del hardware.
- Conclusiones y posibles ampliaciones propuestas.
- Bibliografía, referencias web y recursos empleados en la realización de este proyecto.

La segunda sección recoge la documentación técnica del proyecto y está enfocada desde el punto de vista del desarrollador. Contiene detalles relativos al análisis y diseño del sistema Hortduino divididos en varias secciones:

- Análisis de los requisitos del sistema, donde se estudian las funciones que ha de tener y los objetivos a cumplir por el sistema. Para ello se hace uso de una representación esquemática, empleando diagramas de casos de uso y listado de requisitos de forma tabulada.

- En la especificación funcional del sistema se agrupan los modelos de datos y procesos presentes en el sistema bajo diagramas de flujo de datos (DFDs) y diagramas de entidad-relación (E-R).

- El diseño del sistema agrupa los diseños de la base de datos y las diversas aplicaciones disponibles para el usuario, describiendo el flujo de trabajo y los procedimientos relativos a cada aplicación del sistema, así como el diseño de las diferentes interfaces de usuario disponibles

- La implementación del sistema ofrece ejemplos del código interno de algunos de los procedimientos implementados en las aplicaciones.

- Las pruebas del sistema recopila la fase de testeo de cada aplicación por separado y del sistema en su conjunto.

Por último, la tercera sección contiene el manual de uso de las diferentes aplicaciones disponibles para el usuario, así como la guía de instalación y configuración del software necesario para su correcto funcionamiento.

2 DESCRIPCIÓN GENERAL DEL PROYECTO

El proyecto **Hortduino** (derivado de las palabras *Hortelano* y *Arduino)* nace de la idea de aplicar las nuevas tecnologías y el concepto del *Internet of Things* **(IoT)** al mundo de la horticultura tradicional a pequeña escala. Los llamados *huertos urbanos*, como actividad ecológica y saludable, han experimentado un crecimiento notable en los últimos años dentro de los núcleos de población mas grandes. Los alimentos cultivados por este método son considerados generalmente de gran calidad y son numerosos los restaurantes de renombre que cultivan sus propias materias primas en huertos propios y especializados para ofrecer una experiencia única a sus clientes.

El presente Proyecto Fin de Carrera está dedicado a desarrollar un sistema de monitorización y control de invernadero apoyado en la popular plataforma de hardware libre Arduino y en tecnologías web. Este sistema, compuesto por tres aplicaciones diferenciadas, recibirá datos acerca de las principales variables implicadas en el crecimiento de los cultivos, como pueden ser la humedad del suelo, humedad ambiental y temperatura, obtenidos mediante una red de sensores colocados en el invernadero.

La información enviada desde los sensores será procesada por la aplicación de escritorio, la cual se encargará de su almacenamiento en una base de datos externa y de la automatización y control del riego y ventilación mediante sendos actuadores mecánicos. Esta aplicación de escritorio también será capaz de mostrar los datos recibidos por red desde los sensores en tiempo real y establecer las condiciones máximas y mínimas que se han de mantener para cada variable, alertando al usuario si se encuentran fuera de rango. El usuario podrá seleccionar el modo automático o manual para el control del riego y la ventilación, de modo que se active el sistema de riego si la humedad del suelo cae por debajo de un determinado valor mínimo aceptable o se accione el sistema de ventilación si la temperatura o humedad interior superan un límite máximo.

Para apoyar a la aplicación de escritorio, la aplicación móvil será una réplica en versión reducida de las funcionalidades e interfaz de la aplicación de escritorio, contando con las ventajas de movilidad que ofrecen los dispositivos *ANDROID*. A diferencia de la aplicación de escritorio, la aplicación móvil no tendrá la funcionalidad de incorporar los datos recibidos a la base de datos externa, pero si podrá ajustar los rangos aceptables de las variables monitorizadas y activar/desactivar los controles de riego y ventilación.

Por último, se dispondrá de una página web donde el usuario podrá visualizar gráficamente la información almacenada en la base de datos por la aplicación principal y tener acceso a estadísticas y gráficos históricos de la evolución de las condiciones ambientales del cultivo durante diferentes periodos temporales. En dicha página web se mostrará de forma dinámica la última lectura recibida en los 30 segundos anteriores, el estado de las alertas y sistemas de riego y ventilación, el consumo de agua para riego y el porcentaje restante disponible en el depósito.

A continuación se muestra de forma gráfica el diseño del sistema en su conjunto y las relaciones entre las diferentes partes en una serie de ilustraciones, las cuales son resultantes de la fase inicial de diseño.

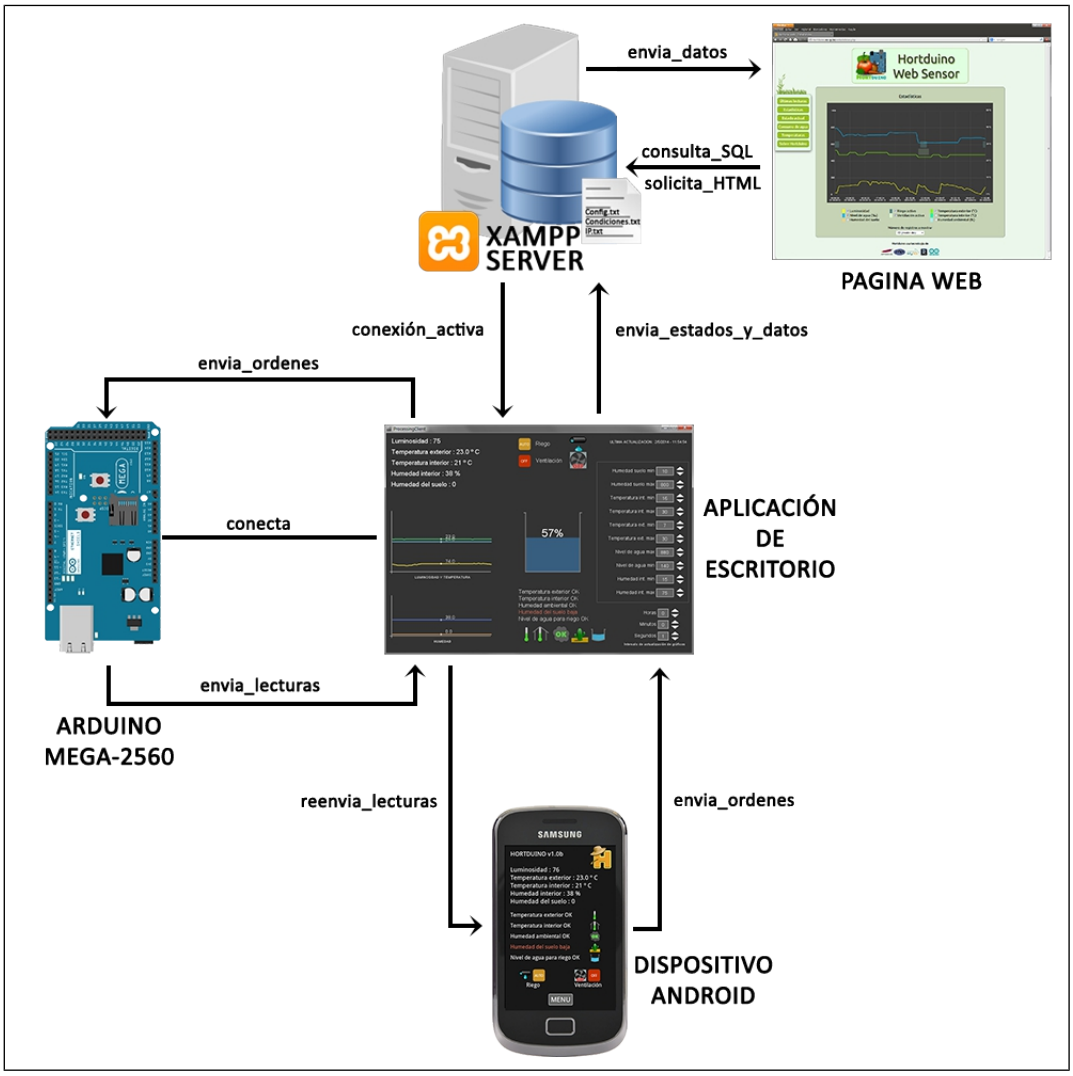


Ilustración 1 - Vista general del sistema y sus componentes

En la anterior ilustración, la vista general del sistema, se pueden observar cómo se relacionan entre sí las diferentes partes de las que se compone el sistema. Seguidamente, veremos en detalle cada relación de forma mas específica y pormenorizada.

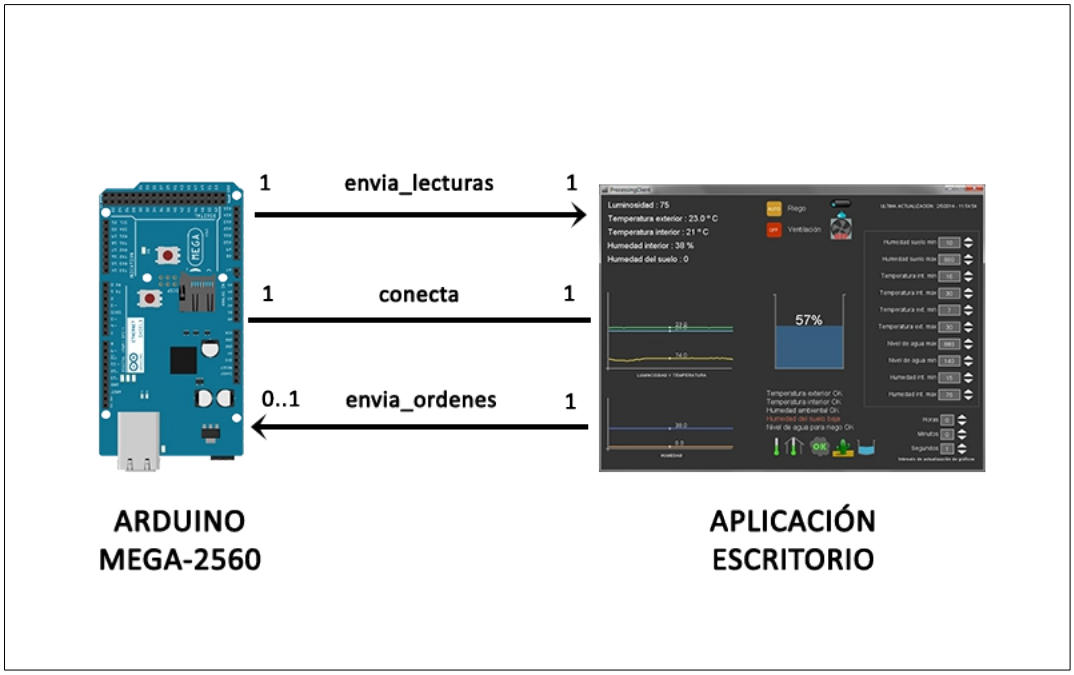


Ilustración 2 – Modelo de negocio Arduino / Aplicación escritorio

En la ilustración 2 se aprecia con mas detalle la relación entre el dispositivo Arduino MEGA-2560 y la aplicación de escritorio. El hardware envía la información recibida desde los diferentes sensores mediante un paquete UDP transferido por red hasta la aplicación de escritorio, la cual es la responsable de procesar la información recibida y actuar en base a los parámetros establecidos por el usuario. Esta aplicación envía un paquete UDP con las órdenes a ejecutar por el hardware, como por ejemplo activar el sistema de riego, cuando las condiciones establecidas o el usuario así lo indiquen.

Hay que destacar que, antes del envío o recepción de paquetes UDP, se debe establecer una conexión entre estos dos elementos. Si no es posible esta conexión, la aplicación de escritorio mostrará un mensaje de advertencia y el menú de configuración de conexión (IP/puerto). El hardware Arduino, provisto de un shield Ethernet para su comunicación en red, solamente enviará los datos si se ha establecido dicha conexión.

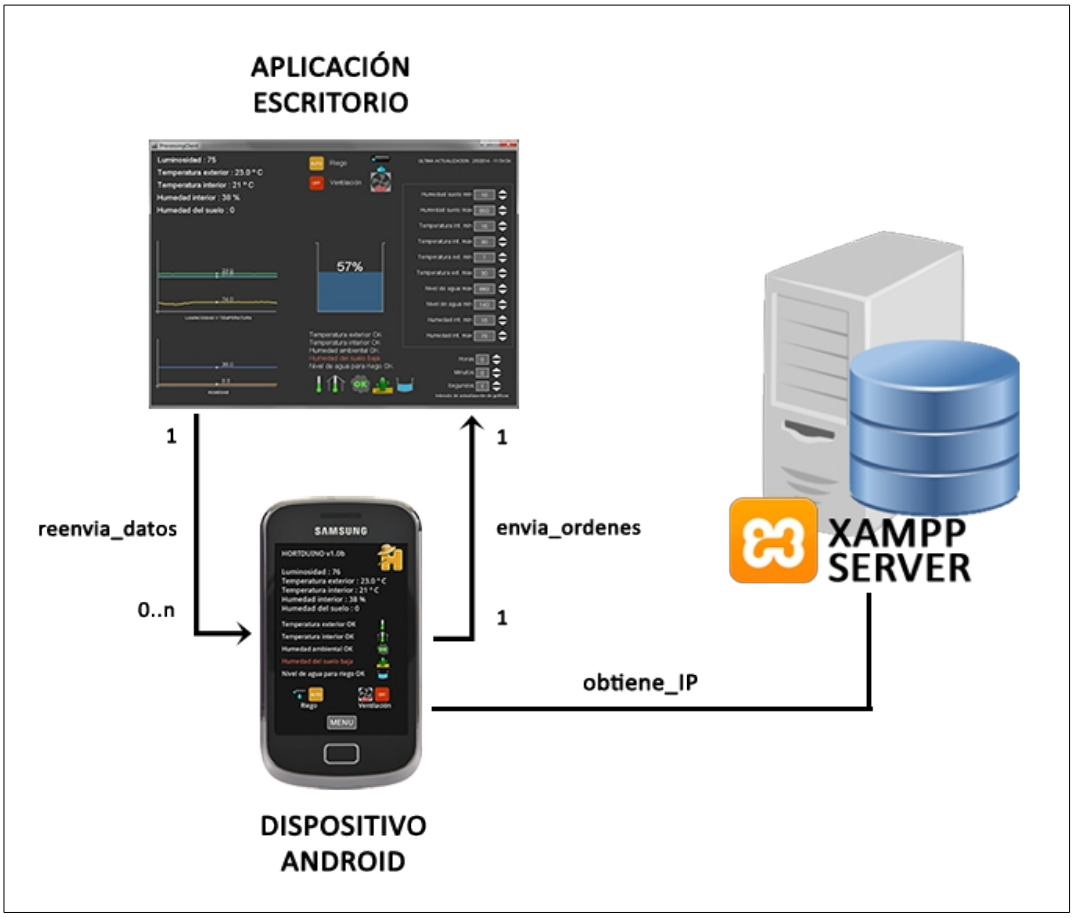


Ilustración 3 – Modelo de negocio Aplicación escritorio / Dispositivo Android

Como podemos observar en la Ilustración 3, el dispositivo Android recibe los datos procesados por la aplicación de escritorio, la cual actúa en modo de *data-relaying,* reenviando la información recibida desde el dispositivo Arduino hacia el cliente Android conectado a la aplicación de escritorio. A su vez, este cliente es capaz de establecer remotamente en la aplicación de escritorio los rangos aceptables de las variables monitorizadas y cambiar el estado de los sistemas de riego y ventilación a voluntad del usuario. Todas las ordenes recibidas en la aplicación de escritorio son transmitidas en el siguiente intervalo temporal disponible hacia el hardware Arduino de modo que sean ejecutadas lo antes posible.

La aplicación que se ejecuta en el dispositivo Android debe conocer la IP de la aplicación de escritorio para recibir la información procesada. Ésto se consigue ejecutando un script PHP en el servidor que recaba la IP asignada en ese momento y la almacena en un fichero de texto plano. El contenido del fichero es leído por la aplicación Android durante su inicialización.

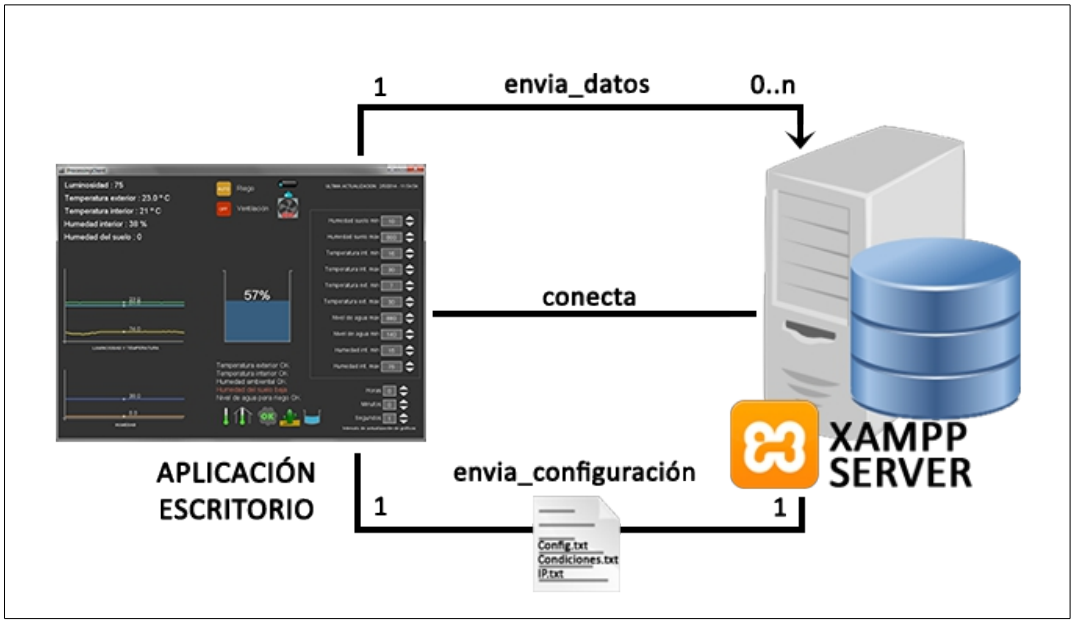


Ilustración 4 – Modelo de negocio Aplicación escritorio / XAMPP Server

La interacción entre la aplicación de escritorio y el servidor XAMPP se efectúa fundamentalmente mediante el envío de datos y el intercambio de la configuración almacenada en los archivos dispuestos para tal fin. Los datos enviados por la aplicación de escritorio son incluidos en la estructura de la consulta SQL que se ejecuta en el módulo MySQL del servidor, incorporándose éstos a la base de datos como resultado de dicha consulta. El envío de datos se establece inicialmente en un intervalo temporal adecuado (10 minutos) para no crear una base de datos demasiado grande, aunque este parámetro puede ser adaptado en el código si se desea un numero mayor o menor de lecturas en un determinado periodo.

Cabe destacar que es necesario establecer una conexión con el servidor para que la aplicación principal pueda inicializarse, ya que la configuración de las variables y el estado de los sistemas de riego y ventilación se almacenan remotamente en dicho servidor.

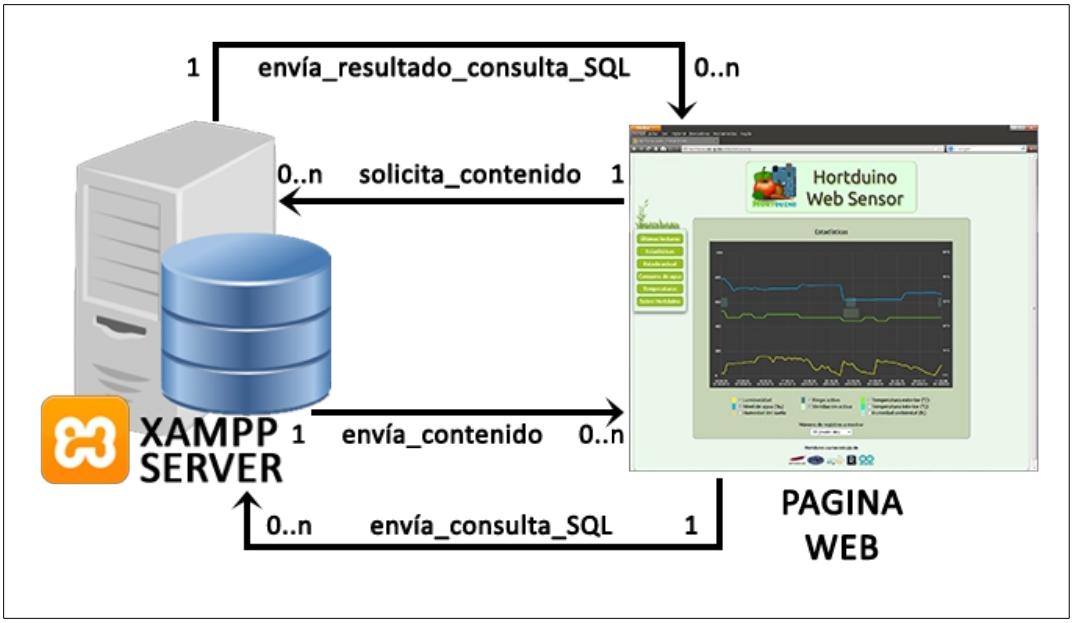


Ilustración 5 - Modelo de negocio XAMPP Server / Web

El la ilustración número 5 se detalla la relación entre el servidor XAMPP y la página web, donde el usuario puede consultar desde un navegador el estado actual del sistema y datos referentes a las variables monitorizadas. El servidor XAMPP dispone de servicios Apache, MySQL e intérprete PHP, cuya combinación genera una página web dinámica a petición del usuario en la que éste puede seleccionar los datos a visualizar de forma gráfica.

Para la representación visual de los datos en pantalla se usan *sketches* programados en *Processing.js*, una adaptación del lenguaje *Processing* a JavaScript destinada a crear elementos incrustables en páginas web usando la tecnología HTML 5. Estos *sketches* son capaces de generar una consulta SQL en el servidor ejecutando una llamada a un script PHP a partir de los parámetros seleccionados por el usuario y mostrar el resultado de forma gráfica.

El navegador web del usuario realiza peticiones de contenido al servidor, donde son atendidas por el módulo Apache y el intérprete PHP, enviando la información requerida para mostrar la página web. Al emplear el lenguaje PHP, se puede variar el contenido mostrado de forma dinámica en función, por ejemplo, del estado de las alertas del sistema para mantener al usuario informado en todo momento de las condiciones y estado actual del invernadero.

2.1 OBJETIVOS DEL PFC

Como ya se ha indicado anteriormente, el objetivo del presente PFC es diseñar e implementar un sistema de control de invernadero apoyado en la plataforma de hardware Arduino. El sistema ha de ser capaz de controlar los sistemas de riego y ventilación del invernadero de forma remota mediante el envío de órdenes, ya sean manuales o como respuesta automática a un cambio en las condiciones establecidas por el usuario. Mas concretamente, el sistema a implementar ha de ser capaz de cumplir estos objetivos:

- Comunicación con el usuario a través de varias herramientas disponibles, algunas de ellas capaces de interactuar de forma remota con el sistema.
- Automatización de los sistemas de riego y ventilación en base a rangos aceptables de algunas variables monitorizadas.
- Almacenamiento de los datos obtenidos a través de los sensores del sistema en una base de datos MySQL para su tratamiento posterior.
- Presentación gráfica de los datos almacenados para su visualización vía web.

2.2 CUESTIONES METODOLÓGICAS

Este proyecto seguirá una metodología de desarrollo basada en sucesivas iteraciones sobre un prototipo inicial, añadiendo nuevas funcionalidades a medida que éstas son testeadas y ampliadas. El empleo de esta metodología permite apreciar la funcionalidad básica del software en desarrollo sin tener en cuenta todas las características del mismo y evaluar el producto de forma temprana por parte de los usuarios, además de observar si se ajusta a los requisitos establecidos inicialmente.

La elección de la metodología basada en prototipos para este proyecto se fundamenta en la efectividad de esta metodología en proyectos con alta interacción con el usuario y su validez en sistemas transaccionales, como los usados en bases de datos, donde las operaciones han de completarse enteramente para ser validadas correctamente. En el caso particular de este proyecto, las dos situaciones anteriores están presentes de forma notable tanto en la parte hardware como en el software, con lo que el ajuste a esta metodología parece ser adecuado para este PFC.

Dentro de los modelos englobados en esta metodología, se usa el Modelo Evolucionario de Prototipado, basado en el sucesivo refinamiento del prototipo inicial. Este modelo tiene como ventajas el poder cambiar o añadir nuevas características no concebidas en la fase inicial de requisitos, haciendo que el producto final se adecúe a las necesidades del usuario y permitiendo que el desarrollador pueda centrarse primero en las partes del sistema que mejor conoce en vez de trabajar sobre el sistema entero.

El desarrollo del software en este proyecto se divide en sucesivas iteraciones o bloques temporales, en las cuales se diseñan y añaden funcionalidades, mejorando si es necesario las ya existentes. Para cada iteración se ejecutan baterías de pruebas a fin de comprobar su funcionalidad. A continuación se detallan las iteraciones ocurridas a lo largo del desarrollo del software:

1 – Estudio previo

En la primera fase se recaba información acerca de la plataforma hardware Arduino y sus herramientas de desarrollo, así como del entorno de programación propio. También se propone un posible inventario con los componentes requeridos para la base hardware del proyecto.

2 – Iteración I

La primera iteración consiste básicamente en conectar y enviar información desde el hardware Arduino hacia el equipo donde se reciben los datos. Esta comunicación se produce mediante el uso de un puerto serial, ya que es la forma mas básica de envío de datos que posee la plataforma Arduino.

3 – Iteración II

En esta iteración se trata fundamentalmente el diseño de la interfaz de usuario para la aplicación de escritorio y la visualización de datos en pantalla usando el lenguaje *Processing*. También incluye la implementación de la comunicación con Arduino mediante paquetes UDP vía Ethernet y el diseño del protocolo de órdenes entre la aplicación y el hardware.

4 – Iteración III

En la tercera iteración se realiza el proceso de portar la aplicación de escritorio al entorno *Android*, aprovechando las facilidades que posee para ello el lenguaje *Processing*. Hay que tener en cuenta que la aplicación resultante no usa la API de *Android* salvo para métodos puntuales, por lo que la interfaz tiene que ser adaptada específicamente a la resolución del dispositivo.

5 – Iteración IV

El diseño de la base de datos y la integración de la librería *BezierSQLib* para *Processing* ocupan la mayor parte de esta iteración. Además se despliega y configura el servidor XAMPP en el PC que hace las veces de servidor.

6 – Iteración V

En la quinta iteración se realiza el diseño y maquetación de la página web del proyecto mediante el uso de los lenguajes HTML, PHP y CSS. Los applets o *sketches* mostrados en la web son implementados también dentro de esta iteración.

7 – Documentación

Durante el desarrollo del proyecto, la mayor parte de los esquemas, notas y borradores obtenidos son incorporados a la documentación en las fases finales del proyecto. Los comentarios en el código son añadidos a la vez que se prueban las diferentes partes del software.

2.3 TECNOLOGÍAS DE DESARROLLO

Para el desarrollo del proyecto Hortduino se han utilizado varias herramientas y tecnologías de uso público y gratuito, siguiendo la filosofía del software libre.

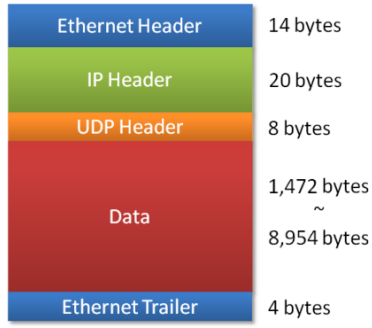
El microcontrolador ATmega2560 instalado en el Arduino se ha programado usando el entorno de desarrollo propio de Arduino, el cual está escrito en *Java* y basado en otros lenguajes similares como *Processing*, el compilador *avr-gcc* y otros programas open-source.

La aplicación de escritorio está programada en *Processing*, un lenguaje similar a *Java* y elegido por su buena capacidad de interacción con el hardware Arduino y por ofrecer la posibilidad de migrar las aplicaciones al entorno *Android* de manera sencilla pero con algunas limitaciones. La interfaz de usuario ha sido diseñada a partir de las herramientas que ofrece *Processing* para mostrar elementos gráficos en pantalla.

En la construcción de la página web se ha utilizado HTML en su versión 5, PHP para la generación dinámica de contenido y *Processing.js* empleado en los applets o *sketches* encargados de representar gráficamente los datos. La maquetación y diseño visual de la web se separa del contenido de la misma a través del uso del archivo CSS (hoja de estilos en cascada) asociado a la página web. Los servicios correspondientes a estas herramientas y el motor SQL de la base de datos son ejecutados en el servidor, agrupados bajo el paquete de desarrollo web XAMPP.

La comunicación entre el dispositivo Arduino, la aplicación de escritorio y la aplicación Android utiliza el protocolo UDP sobre Ethernet mediante la biblioteca UDP *Hypermedia.net* para *Processing* y *EthernetUDP.h* para Arduino. Se ha elegido este protocolo de red por su buen ajuste a las características técnicas de este proyecto, siendo las principales expuestas a continuación:

- ✔ Intercambio rápido de información usando un solo paquete UDP, lo que evita el establecimiento de una conexión previa (*handshaking*) y la sobrecarga asociada al uso del protocolo TCP (20 bytes para el header en TCP, 8 bytes usados en el header UDP), con el correspondiente ahorro en el tráfico de datos enviados por la red.
- ✔ La pérdida de un paquete en la red no supone una problema crítico, puesto que el envío de datos y órdenes se realiza de forma casi continua. En vez de solicitar el reenvío si se emplease el protocolo TCP, los datos recibidos en el servidor son reemplazados por la siguiente lectura con valores prácticamente idénticos a los del paquete perdido. En el caso del envío de órdenes se presenta una situación similar, ya que son enviadas al hardware de forma continuada.



Estructura interna de un paquete UDP

En lo que respecta al soporte hardware del proyecto, se han empleado los siguientes elementos en la construcción del prototipo hardware:

- Placa Arduino MEGA 2560 R3 con microcontrolador ATMega integrado.
- Shield Ethernet compatible con Arduino.
- Sensor de temperatura y humedad DHT11.
- Sensor de temperatura National Semiconductor LM35.
- Sensor de humedad del suelo (de fabricación propia).
- Sensor de medición de distancia por ultrasonidos.
- Fotorresistencia mini.
- Relés, válvula de solenoide y ventilador 12V.
- Cableado, breadboard y resistencias eléctricas.

Para el desarrollo y pruebas del software, han sido necesarios los recursos y herramientas enumerados a continuación:

- PC dedicado a tareas de programación y servidor.
- Smartphone Samsung Galaxy Mini II con Android 2.3 para el desarrollo de la aplicación Android y las correspondientes pruebas.
- Router TP-LINK TD-W8970 Wi-Fi para las comunicaciones por red.
- Editor de código Sublime Text 2 (versión de prueba).
- Entorno de desarrollo Arduino 1.0.5
- Entorno de desarrollo Processing 2.1.1
- Java JDK 1.7.0.45 y Android SDK Tools.
- Servicio de redireccionamiento DNS *No-IP.org* (gratuito).
- Fritzing 0.8.7 usado en el diseño del prototipo hardware.
- LibreOffice 4.2 para labores de documentación.
- Adobe Photoshop CS6 trial, para la creación de las ilustraciones que acompañan la documentación, diversas imágenes auxiliares en la página web y algunos elementos gráficos de las aplicaciones.

3 DESCRIPCIÓN GENERAL DEL PRODUCTO

Desde el punto de vista relativo al usuario, el sistema resultante se divide en dos grandes partes diferenciadas: el ***back-end***, que comprende todos los procesos no visibles por el usuario, y el ***front-end*** que es el encargado de la interacción con el usuario mediante las interfaces propias de cada herramienta.

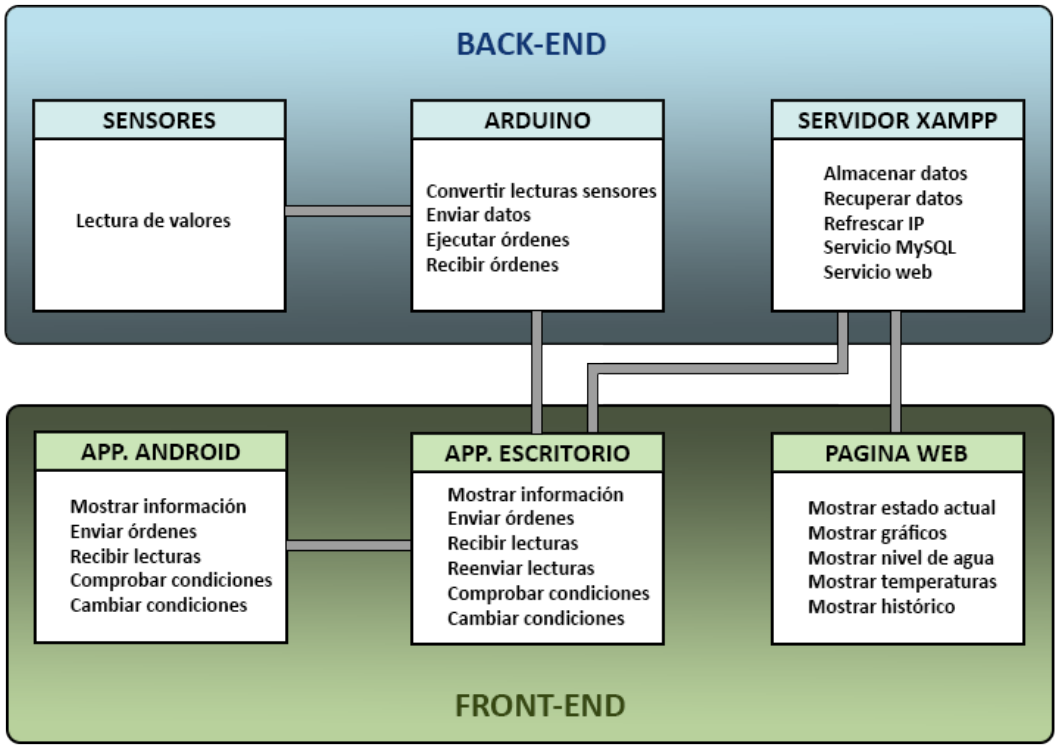


Ilustración 6 – Módulos y funciones del sistema

En la anterior ilustración se puede apreciar la separación en los subsistemas ***back-end*** y ***front-end*** explicada anteriormente y los diferentes módulos que están relacionados entre sí mediante procesos de comunicación y/o intercambio de datos. Dado que el presente proyecto poseé varios subsistemas diferenciados, vamos a tratar cada una de ellos por separado a la hora de describir su funcionalidad y composición.

3.1 *FRONT-END*

3.1.1 Aplicación de Escritorio

Descripción y funcionalidad

Esta aplicación se podría considerar como el corazón del sistema, puesto que se encarga de las funciones principales, como son la comunicación directa con el hardware Arduino y la interacción con el usuario. Desde esta aplicación se controlan los rangos aceptables de las variables monitorizadas (humedad ambiental, humedad del suelo suelo, temperatura interior, temperatura exterior y nivel de agua para riego) así como el modo de funcionamiento de los sistemas de riego y ventilación (ON, OFF, AUTO). Estas condiciones y el estado actual de los sistemas de riego y ventilación son almacenados en sendos ficheros de texto dentro del servidor XAMPP cada vez que se modifican los parámetros en la aplicación para mantener la consistencia de configuraciones y estados entre los subsistemas *front-end*. La interfaz se ha tratado de simplificar todo lo posible para hacerla intuitiva y sencilla a la vez que funcional. En el manual de usuario se detalla mas en profundidad las distintas partes y controles disponibles en la interfaz de la aplicación.

Interacción con otros sistemas

La aplicación de escritorio se ocupa de enviar la información a almacenar en la base de datos del servidor XAMPP y, si existe una conexión activa desde el dispositivo Android, reenviar la información recibida en tiempo real hacia el dispositivo conectado y ejecutar las órdenes recibidas desde el mismo en el hardware Arduino.

Limitaciones y dependencias

Hay que destacar que es indispensable para el inicio de la aplicación de escritorio que los servicios del servidor web Apache y el motor MySQL del servidor XAMPP se encuentren en ejecución, ya que es necesario establecer la dirección IP donde se encuentra conectado el hardware Arduino para la interacción con el mismo. De otro modo, la aplicación fallará al arrancar.

Al estar basada en *Java*, es necesario que se encuentre instalado el *Entorno de Tiempo de Ejecución* (Java Runtime Environment) de *Java* en el equipo donde se ejecute esta aplicación y que se disponga de conexión a la red Ethernet en la cual se encuentra conectado el hardware del sistema.

Usuarios

Esta aplicación está orientada a los administradores / encargados del invernadero, pues es la pieza básica del sistema Hortduino en lo referido al software.

3.1.2 Aplicación para dispositivos Android

Descripción y funcionalidad

La aplicación para dispositivos Android se concibió como un complemento a la aplicación de escritorio, dado el enorme número de dispositivos con dicho sistema operativo presentes en el mercado. En esta aplicación se replican las funcionalidades principales de la aplicación de escritorio en un formato compacto y adaptado a la entrada táctil. En la sección correspondiente a esta aplicación dentro del manual de usuario se aborda su uso y disposición de los elementos de la interfaz.

La característica fundamental de esta aplicación se la otorga el uso del sistema operativo Android y los dispositivos en los que se puede ejecutar, y no es otra que la movilidad que confiere al usuario mientras hace uso de las funciones de la aplicación de forma remota.

Interacción con otros sistemas

En lo relativo a la comunicación con otros subsistemas, esta aplicación interactúa con la aplicación de escritorio, siendo esta última la encargada de ejecutar las órdenes enviadas desde la aplicación Android y de reenviar la información obtenida del hardware Arduino hacia el cliente Android. También interactúa con el servidor XAMPP para cargar la configuración que establece el usuario.

Esta aplicación tiene la misma prioridad que la aplicación de escritorio en lo concerniente al cambio de parámetros de los rangos aceptables pertenecientes a las variables monitorizadas y al envío de órdenes a los sistemas de riego y ventilación.

Limitaciones y dependencias

A diferencia de la aplicación de escritorio, la aplicación para Android no poseé funcionalidad de comunicación con la base de datos usada para almacenar la información recibida y, al igual que la aplicación de escritorio, necesita que ésta última y los servicios del servidor XAMPP se encuentren en ejecución. Si no se cumplen estos requisitos, la aplicación para Android no se iniciará. Es necesario que el dispositivo Android disponga de la versión 2.3.6 o superior de dicho sistema operativo.

Otra limitación a destacar está relacionada con el uso del lenguaje *Processing* para la conversión de la aplicación de escritorio hacia el entorno Android: al no usar el API de Android, la interfaz de la aplicación ha de adaptarse por medios propios a la resolución y densidad de la pantalla del dispositivo que la ejecute.

Usuarios

Al ser una aplicación prácticamente idéntica a la aplicación de escritorio en lo que respecta a funcionalidad y apariencia, este software está orientado al administrador / encargado del invernadero.

3.1.3 Página Web

Descripción y funcionalidad

El principal cometido de la página web es el de servir como herramienta de observación sobre la evolución en el tiempo de los parámetros relativos al invernadero controlados por el sistema Hortduino y actuar como plataforma para acceder a la información almacenada en la base de datos. De esta manera, el usuario puede visualizar la información de manera gráfica y valorar el desarrollo de los cultivos en función de las condiciones establecidas en el sistema.

Al ser una página web creada de forma dinámica mediante el lenguaje *PHP*, se ofrece al usuario la posibilidad de comprobar desde cualquier lugar donde se disponga de un navegador web y conexión a Internet la información siguiente:

- ✔ Estado actual de los sistemas de riego y ventilación.
- ✔ Estado actual de las condiciones de los parámetros monitorizados, incluyendo posibles alertas.
- ✔ Nivel de agua disponible en el depósito de riego, su duración aproximada con el consumo actual y el total consumido hasta ese momento.
- ✔ Temperaturas máximas, medias y mínimas registradas.
- ✔ El comportamiento de las variables y la actividad de los sistemas de riego y ventilación a lo largo de un intervalo de tiempo seleccionado por el usuario.

Mediante el uso del lenguaje de hojas de estilo en cascada (CSS) y las reglas definidas con éste en el archivo *estilo.css* se separa el contenido de la página del aspecto visual, facilitando así la posible modificación de los elementos visuales que componen la página sin alterar la funcionalidad o el contenido de la misma.

Interacción con otros sistemas

Se puede considerar a la página web como la parte del subsistema *Servidor-XAMPP* que es visible por el usuario y se comunica dentro de ese subsistema con los servicios MySQL y servidor web. Dentro del directorio de la página web en el servidor se alojan tres ficheros de texto plano utilizados para almacenar configuraciones y estados. Estos ficheros son:

- *Config.txt* : Contiene los valores mínimos y máximos aceptables para cada variable monitorizada, separados por punto y coma. Usado por la aplicación de escritorio.
- *Condiciones.txt* : Almacena el estado de las condiciones del invernadero en formato numérico usando separación por comas mediante el código **0** si el valor actual de dicha condición se encuentra dentro del rango fijado por el usuario, **-1** si el valor actual es inferior al mínimo establecido y **1** si se supera el máximo permitido para esa condición. Se usa para indicar el estado de las condiciones en la página web.
- *IPServer.txt* : Este archivo es utilizado para actualizar la dirección IP empleada en la inicialización de la aplicación Android. Se modifica su contenido mediante la invocación a un script que la recupera mediante funciones PHP.

Limitaciones y dependencias

Para que la página web pueda ser mostrada en el navegador del usuario, el servidor web Apache y su servicio asociado deben estar en ejecución en el host encargado de actuar como servidor. Igualmente, es crítico que se encuentre en ejecución el motor MySQL para poder realizar consultas a la base de datos y mostrar los datos obtenidos en los *sketches* dispuestos en la web.

Se necesita un navegador que soporte HTML 5 para su correcta visualización.

Usuarios

La página web del proyecto Hortduino está dirigida a cualquier usuario que quiera consultar los datos disponibles contemplados en la sección *Descripción y Funcionalidad* descrita mas arriba. Para ello, el usuario ha de conocer la dirección de la página web, disponer de un navegador y conexión a Internet o bien a la red Ethernet donde se encuentre el servidor XAMPP.

3.2 *BACK-END*

3.2.1 Arduino

Descripción y funcionalidad

El dispositivo Arduino es una plataforma de hardware libre asequible y ampliamente utilizada para toda clase de proyectos que tengan que ver con la presencia de sensores y actuadores mecánicos, así como proyectos de robótica y domótica. En nuestro caso, es el encargado de recibir y transformar la información obtenida en la lectura de los sensores y trasladarla a la aplicación de escritorio por medio de un *shield* Ethernet, proporcionando la conectividad necesaria con el router. Otra función importante que desarrolla este hardware es la de activar y desactivar los actuadores correspondientes a los sistemas de riego y ventilación en base a las órdenes recibidas desde la aplicación de escritorio.

El proyecto Hortduino usa un Arduino modelo *Mega 2560*, dotado de un microcontrolador ATmega 2560. Las características físicas de este hardware son explicadas en mayor detalle dentro del apéndice correspondiente, siendo las mas destacadas:

- Memoria flash : 256KB (8 KB usados por el bootloader)
- Memoria SRAM : 8 KB
- Pines de entrada analógicos : 16
- Pines de entrada/salida digitales: 54
- Velocidad de reloj : 16 Mhz
- Voltaje operativo : 5 V

Interacción con otros sistemas

Los sistemas que interaccionan con el dispositivo Arduino son el array de sensores, de donde se toman las lecturas ambientales, y la aplicación de escritorio, desde la que se reciben las órdenes a ejecutar en los sistemas de riego y ventilación, y hacia la que se envía la información de las lecturas mediante paquetes de datos usando el protocolo de red UDP.

Limitaciones y dependencias

La programación del microcontrolador se efectúa empleando el entorno de desarrollo propio de Arduino para almacenar en la memoria flash del dispositivo el programa a ejecutar, el cual es transferido al dispositivo por medio de un cable USB. Dado que la memoria SRAM disponible es limitada, se debe optimizar el código para que se nunca se exceda el tamaño máximo de la memoria usada en tiempo de ejecución. Si esto sucediese, el dispositivo mostraría un ciclo de reinicialización continuo por la falta de memoria disponible para ejecutar las siguientes instrucciones.

El subsistema compuesto por el dispositivo Arduino, los sensores y actuadores mecánicos depende de la aplicación de escritorio para conocer cuándo se debe regular el comportamiento de los sistemas de riego y ventilación o hay indicios de un posible fallo en algún sensor.

El uso de la librería *Ethernet.udp.h* impone un tamaño máximo de datos por paquete UDP de 24 bytes que hay tomar en consideración a la hora de construir el datagrama para no exceder este tamaño.

Usuarios

Puede ser de utilidad al administrador del invernadero conocer cómo está trabajando internamente el dispositivo Arduino en lo que respecta a los datos enviados y a las órdenes recibidas. Esto se consigue mediante el uso de un cliente que soporte el protocolo de comunicación de puerto serie local (como la aplicación *PuTTY*) mediante el cual se puede acceder a los datos que maneja Arduino en modo debug, conectando el dispositivo a un PC a través de un cable USB.

3.2.2 Sensores

Descripción y funcionalidad

El array de sensores es un conjunto de actuadores mecánicos y diversos sensores electrónicos capaces de medir diferentes variables. Cada sensor dispuesto tiene la capacidad de transmitir la medición efectuada de diferentes maneras en función del tipo de sensor. La información técnica referente a cada sensor se incluye en el apéndice correspondiente. A continuación aparecen los diferentes sensores usados en el proyecto:

- National Semiconductor LM35 (temperatura exterior).
- Grove Temperature and Humidity Sensor (temperatura y humedad interior).
- Fotorresistencia CdS (intensidad lumínica).
- Seeed Ultrasonic Sensor (nivel de agua en el depósito de riego).
- Sensor de humedad del suelo (basado en la conductividad del yeso húmedo).

Interacción con otros sistemas

El subsistema de sensores y actuadores solamente se comunica con el dispositivo Arduino mediante los cables conectados a las diferentes entradas y salidas de las que dispone el hardware.

Limitaciones y dependencias

La principal limitación de este subsistema es la precisión que ofrece cada sensor en la medida de la variable de la que se ocupa. En general, los rangos de medida de los sensores son adecuados para el modelo de temperatura y humedad que cabría esperar en el entorno de un invernadero, salvo en el caso de la fotorresistencia empleada en la medición de la luminosidad, la cual posee una precisión baja en sus lecturas (debido a su reducido coste) que no la hace apta si se necesita una precisión aceptable en la medición de la intensidad lumínica.

Usuarios

No hay usuarios que puedan hacer uso de este subsistema.

3.2.3 Servidor XAMPP

Descripción y funcionalidad

Este subsistema está compuesto principalmente por el paquete de desarrollo web XAMPP y el PC donde se ejecuta dicho software. El servidor XAMPP es el encargado de proporcionar los servicios básicos que usan el subsistema de la página web y de la aplicación de escritorio, como son el motor MySQL de la base de datos, el intérprete PHP y el servicio web. También se ocupa de hospedar la base de datos usada en el proyecto para almacenar las lecturas, el contenido y los recursos de la página web y los archivos de configuración mencionados anteriormente en el apartado **3.1.3**.

Interacción con otros sistemas

El servidor XAMPP se relaciona básicamente con dos sistemas: la aplicación de escritorio y la página web, la cual se puede considerar embebida dentro del servidor XAMPP.

La aplicación de escritorio realiza consultas SQL que son atendidas por el servicio MySQL en ejecución en el servidor para incorporar las lecturas recibidas desde el hardware Arduino a la base de datos del servidor. Esta aplicación también solicita la lectura y modificación de los archivos de texto *Config.txt* y *Condiciones.txt* alojados en el servidor para recuperar o guardar, según sea el caso, la configuración de la aplicación usando para ello las funciones contenidas dentro de diversos scripts PHP.

En el caso de la página web, se utiliza el servicio MySQL para recuperar los registros de la base de datos, procesarlos y mostrar la información requerida por el usuario en los diferentes *sketches* dispuestos en la web para tal fin.

Limitaciones y dependencias

La potencia de procesamiento del host sobre el que se ejecuta el servidor XAMPP puede suponer una limitación a la hora de trabajar con una base de datos muy grande o un número elevado de conexiones concurrentes en la página web, factor a tener en cuenta si se desea escalar el proyecto.

En lo respectivo a la dependencia hay que decir que, aunque no es necesario contar con conectividad de red para poder hacer uso de las funciones y herramientas que ofrece el paquete XAMPP, este proyecto se basa en la conexión al servidor mediante Internet o red local para construir un sistema que puede ser controlado y observado de forma remota.

Usuarios

El servidor XAMPP de este proyecto se gestiona por parte del administrador o la persona encargada de la logística del invernadero.

4 ARQUITECTURA DEL PROYECTO

El proyecto Hortduino sigue una arquitectura dirigida por eventos, apoyada en el uso de varias tecnologías que permiten generar una reacción ante un cambio significativo en el estado del sistema. En la siguiente ilustración se muestra un esquema general de la arquitectura utilizada en Hortduino.

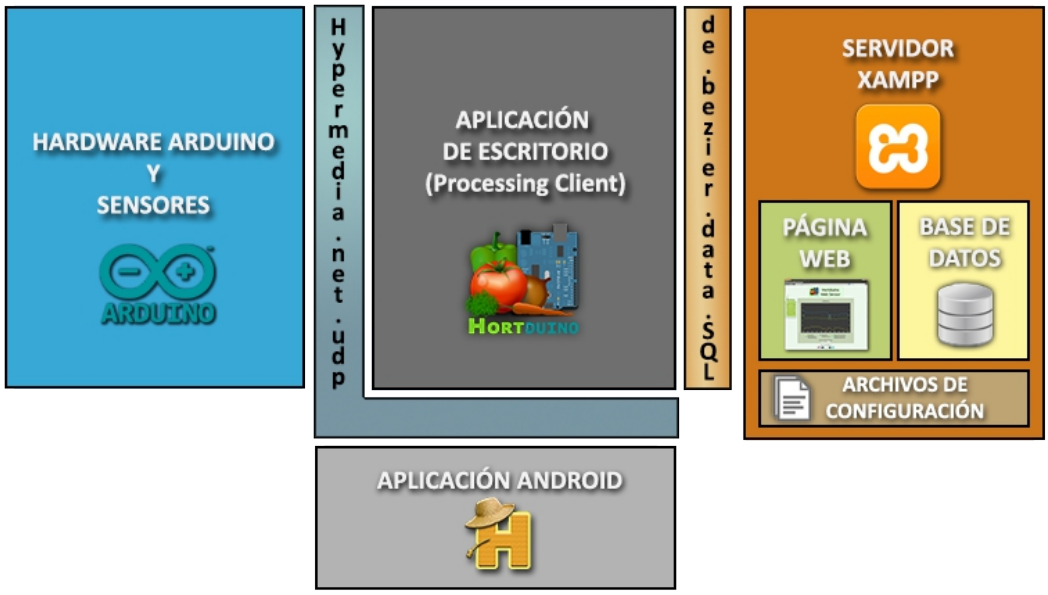


Ilustración 7 – Arquitectura del Proyecto Hortduino

Mediante el uso de la librería *Hypermedia.net.udp* para el entorno de desarrollo de *Processing*, se logra que la aplicación de escritorio se comunique, usando el protocolo de red UDP, con el dispositivo Arduino y la aplicación para dispositivos Android. A su vez, el dispositivo Arduino emplea la librería *Ethernet.udp* con similar funcionalidad que la anterior pero adaptada al lenguaje de programación del microcontrolador que posee Arduino.

El empleo de otra librería para el entorno *Processing* es la encargada de que las consultas SQL para añadir los datos registrados por la aplicación de escritorio sean ejecutadas por el motor MySQL del servidor XAMPP y se incorporen dichas lecturas a la base de datos alojada en el mismo servidor.

En lo respectivo al servidor XAMPP, se encarga de hospedar la página web así como la base de datos utilizada para almacenar las lecturas enviadas desde la aplicación de escritorio. Estas funciones cuentan con sus respectivos servicios en ejecución en el servidor. También provee los ficheros de configuración a las aplicaciones que hacen uso de ellos.

4.1 Arquitectura interna del subsistema Arduino

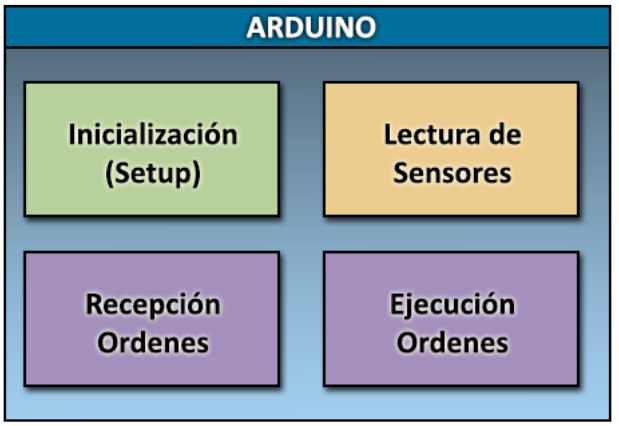


Ilustración 8 – Arquitectura interna del subsistema Arduino

Como se puede apreciar en la ilustración anterior, el subsistema que engloba al dispositivo Arduino y los sensores está compuesto por cuatro módulos que agrupan diferentes procedimientos, encargados de las diferentes tareas que es capaz de llevar a cabo el hardware.

- **Inicialización (Setup)** : Este procedimiento contiene la inicialización del dispositivo Arduino estableciendo los pines de entrada/salida utilizados y los valores iniciales de las variables. Se ejecuta automáticamente al inicio del programa y no puede ser invocado posteriormente.
- **Lectura de sensores** : Agrupa las tareas de conversión y muestreo de las señales recibidas desde los sensores en los intervalos de lectura establecidos y el envío hacia la aplicación de escritorio de estos datos.
- **Recepción Ordenes** : Incluye las funciones de la librería *Ethernet.Udp* que implementa la escucha en un puerto UDP determinado para la recepción del paquete que contiene las órdenes enviadas por la aplicación de escritorio.
- **Ejecución Ordenes** : Implementa la traducción de las órdenes recibidas mediante paquetes UDP a acciones físicas en los actuadores de los sistemas de riego y ventilación si es necesario cambiar su estado.

4.2 Arquitectura interna del subsistema Aplicación de Escritorio

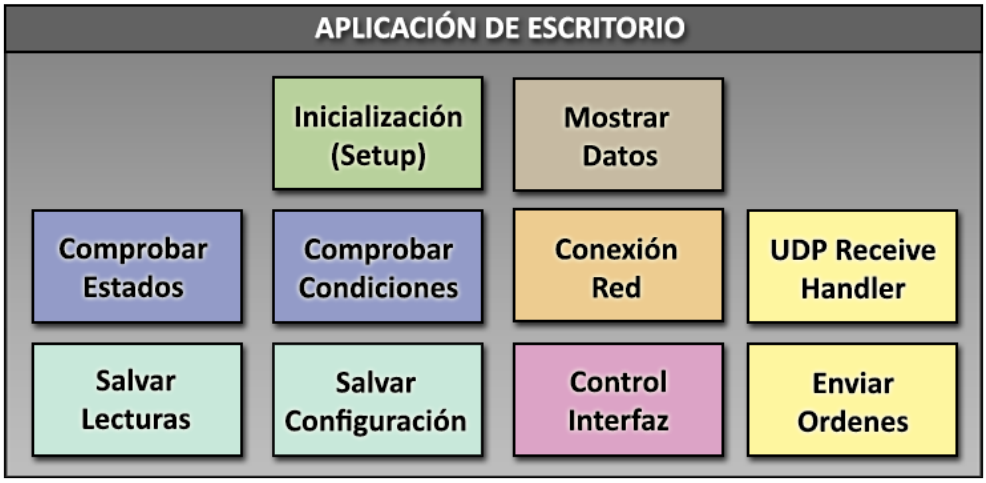


Ilustración 9 – Arquitectura interna de la Aplicación de Escritorio

En la ilustración de mas arriba se observan los bloques funcionales de los que se compone la aplicación para escritorio. Cada bloque representa un grupo de procedimientos relacionados a una tarea concreta. Seguidamente se recoge la descripción y funcionalidad de cada uno de estos conjuntos de procedimientos y funciones.

- **Inicialización (Setup)** : Se ocupa de establecer los valores iniciales de las variables empleadas, resolver la conexión a la base de datos remota y al hardware Arduino y cargar las configuraciones de la aplicación usando los archivos almacenados en el servidor.
- **UDP Receive Handler** : Este módulo básico de la librería *Hypermedia.net.UDP* cumple una doble función: por un lado se mantiene a la escucha de los paquetes UDP entrantes enviados desde el dispositivo Arduino para convertirlos en datos usables por la aplicación, y por otra parte monitoriza la conexión del dispositivo Android a la aplicación para reenviar las lecturas obtenidas y recabar las órdenes enviadas desde dicho dispositivo.
- **Enviar Ordenes** : Este procedimiento se encarga de enviar las órdenes generadas por la aplicación de escritorio al dispositivo Arduino según el intervalo de tiempo fijado. También se ocupa de recargar las condiciones para reflejar los posibles cambios de estado si es necesario.
- **Comprobar Estados** : Se trata de los procedimientos implicados en actualizar el estado de los sistemas de riego y ventilación en base al modo de funcionamiento establecido por el usuario mediante los botones correspondientes en la interfaz.

- **Comprobar Condiciones** : Comprende el procedimiento que compara los valores actuales de las variables monitorizadas con los rangos fijados por el usuario de la aplicación. Se encarga de informar al usuario activando alertas visuales en la interfaz de la aplicación si hay valores fuera del rango nominal establecido para cada condición.
- **Conexión Red** : Es el encargado de comprobar si la conexión con el dispositivo Arduino es funcional. En caso de que esta conexión no fuera posible o se presentase una interrupción en la misma, este módulo se encarga de indicarlo en la interfaz de la aplicación y mostraría un menú para configurar los parámetros IP y puerto a los que conectar.
- **Salvar Lecturas** : Mediante el empleo de la librería *de.bezier.data.SQL*, este procedimiento se ocupa de construir y enviar una consulta en lenguaje SQL al servicio MySQL del servidor XAMPP para la inclusión de los datos contenidos dentro de esa consulta en la base de datos. Esta acción se realiza en intervalos de tiempo fijados.
- **Mostrar Datos** : Cumple la función de mostrar los datos obtenidos por red en tiempo real sobre las variables supervisadas, así como de informar al usuario de la hora y fecha en la que fue recibida la última lectura.
- **Control Interfaz** : Es un conjunto de procedimientos que se ocupan de mostrar y actualizar en la interfaz de la aplicación diferentes controles y elementos informativos para el usuario, como pueden ser:
    - Gráficos representativos de la evolución temporal sobre algunas de las variables supervisadas (Humedad ambiental y del suelo, luminosidad, temperatura interior y exterior) así como los controles para alterar el intervalo de tiempo transcurrido entre cada actualización.
    - Nivel actual en porcentaje del depósito de agua para riego.
    - Valores fijados de los rangos escogidos para cada variable y sus correspondientes selectores para modificarlos.
    - Estados y alertas de las condiciones actuales registradas.
    - Botones para el cambio de modo de control en los sistemas de riego y ventilación del invernadero así como animaciones para informar del estado actual de esos sistemas (activado o desactivado).
- **Salvar Configuración** : Este procedimiento guarda la configuración de la aplicación remotamente en el servidor si se produce algún cambio en los valores mínimos y máximos aceptables de las variables o si se cambia el modo de actuación de los sistemas de riego y ventilación. Dada la limitación del lenguaje *Processing* a la hora alterar el contenido de un archivo remoto, se ha optado por el empleo de scripts en PHP que realicen esta función al ser invocados desde la aplicación de escritorio.

4.3 Arquitectura interna del subsistema Aplicación Android

Ilustración 10 – Arquitectura interna de la Aplicación para Android

Como ya se ha indicado anteriormente, la aplicación para dispositivos Android es una adaptación de la aplicación de escritorio al entorno Android pero con algunas limitaciones y funcionalidades que la hacen ligeramente diferente de la aplicación de escritorio. La arquitectura de este subsistema se detalla a continuación.

- **Inicialización (Setup)** : Procedimiento que se encarga de establecer las variables del programa a los valores iniciales necesarios, poner en marcha el *socket* UDP de escucha para recibir datos y enviar órdenes y finalmente cargar la configuración de la aplicación guardada en el servidor XAMPP.
- **Control TouchEvent** : Simula el comportamiento del ratón mediante eventos de entrada táctil de la librería *android.view.MotionEvent*.
- **UDP Receive Handler** : Este módulo es similar al empleado por la aplicación de escritorio. Realiza funciones de recepción de datos y su posterior conversión para ser usados en la interfaz de la aplicación.
- **Salvar Configuración** : Al igual que en la aplicación de escritorio, es el procedimiento encargado de guardar remotamente la configuración de la aplicación en el servidor XAMPP cuando ocurre un cambio significativo en ella.

- **Mostrar Menú** : Es el conjunto de procedimientos que generan un menú en pantalla con varias opciones para el usuario.
- **Mostrar Nivel Depósito** : Se encarga de dibujar una representación del depósito de agua para riego y su porcentaje de llenado actual en la pantalla del dispositivo.
- **Mostrar Inicio** : Se ocupa de mostrar la pantalla inicial de la aplicación con información relativa a los datos recibidos en tiempo real, el estado nominal de las condiciones del invernadero, las posibles alertas y el estado y funcionamiento de los sistemas de riego y ventilación, todo ello de forma gráfica y compacta en la pantalla del dispositivo. Se apoya en la clase `Botón` implementada en la aplicación para mostrar los diferentes botones accesibles por el usuario, los cuales son capaces de cambiar el modo de funcionamiento del riego y la ventilación del invernadero y acceder al menú de la aplicación.
- **Mostrar selectores** : Este conjunto de procedimientos dibuja los *sliders* sobre los cuales el usuario puede fijar el valor máximo y mínimo aceptable para cada variable monitorizada por el sistema Hortduino. Para ello hace uso de la clase propia `Selector` que se ha implementado específicamente con este fin.
- **Comprobar Estados** : Comprueba el estado del sistema de riego y ventilación del invernadero, actuando de acuerdo al modo de funcionamiento establecido por el usuario. También muestra la actividad de dichos sistemas de forma visual.
- **Comprobar Condiciones** : Son los procedimientos encargados de alertar al usuario si alguna de las condiciones establecidas sobre los parámetros supervisados por el sistema Hortduino se encuentra fuera de los valores permisibles fijados por el usuario.
- **Conexión Red** : Es el procedimiento que se ocupa de alertar si la conexión con el servidor XAMPP y la aplicación de escritorio no se ha podido iniciar o ha sufrido una interrupción.
- **Enviar Ordenes** : Se encarga de enviar las nuevas órdenes por red a la aplicación de escritorio si se detecta un cambio en las condiciones establecidas por parte del usuario. Hay que destacar que las órdenes enviadas desde esta aplicación tienen prioridad sobre las relativas a la aplicación de escritorio. Esto se hace para evitar órdenes contradictorias que afectarían al normal funcionamiento del sistema.

4.4 Arquitectura interna del subsistema Servidor XAMPP

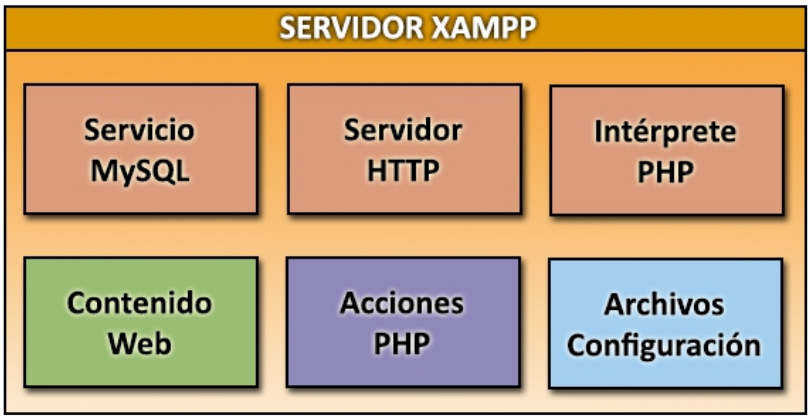


Ilustración 11 – Arquitectura interna del Servidor XAMPP

El servidor XAMPP es un conjunto de servicios y herramientas formado por el sistema de gestión de bases de datos *MySQL*, el servidor web *Apache* y los intérpretes para lenguajes *PHP* y *Perl*, todo ello bajo un mismo paquete que se instala en el PC destinado al rol de servidor. En el caso del proyecto Hortduino, es el encargado de mantener la base de datos y los servicios que son necesarios para que otros subsistemas puedan inicializarse y operar correctamente, hospedar el contenido de la página web y almacenar los archivos de configuración que mantienen la coherencia interna entre los diferentes subsistemas.

- **Servicio MySQL** : Este módulo está compuesto por los servicios necesarios para operar con la base de datos y se encarga de tareas como la realización de consultas o el soporte de conexiones de usuarios a la base de datos.
- **Servidor HTTP** : Se trata del servicio encargado de responder a las peticiones HTTP efectuadas desde el navegador web del usuario solicitando el envío del contenido y estructura de la página web.
- **Intérprete PHP** : Su función es la de traducir y ejecutar en el servidor el código PHP de la página web a fin de generar dinámicamente un documento HTML, con el resultado del procesamiento del archivo PHP, que será enviado por el servidor HTTP al navegador web del usuario.
- **Contenido Web** : Aquí se agrupan todos los recursos requeridos para mostrar la página web y su estructura en el navegador del usuario, como pueden ser imágenes, archivos *.php* correspondientes a cada sección de la web, *applets* y el archivo *.css* de estilos visuales de la página.

- **Acciones PHP** : Es el conjunto de archivos usados por diferentes procedimientos para realizar acciones remotamente en el servidor mediante el empleo de los scripts contenidos en ellos. A continuación se listan dichos archivos y sus funciones:
  - *AccionesConfig.php* : Es usado por la aplicación de escritorio para guardar y cargar la configuración de las variables supervisadas almacenadas en el archivo de texto plano *config.txt*. También actualiza el estado de las condiciones en el archivo de texto *condiciones.txt.*
  - *Consultar.php* : Este script genera una consulta SQL a la base de datos del servidor y devuelve el resultado al procedimiento que la solicitó. Se emplea en varios *sketches* de la página web como medio de acceso al contenido de la base de datos.
  - *GetIP.php* : Devuelve la actual dirección IP del servidor. Es necesario para la inicialización de las aplicaciones Escritorio y Android.

- **Archivos de configuración** : Comprende los archivos de texto plano vistos anteriormente en la sección **3.1.3** y su apartado **Interacción con otros sistemas**.

El paquete XAMPP incluye en su instalación un panel de control desde el cual se administra y configura el funcionamiento de los diversos servicios que integra. Esta cuestión se aborda en la sección correspondiente del manual de usuario para mayor claridad.

5 DESPLIEGUE DEL PRODUCTO

El sistema de control de invernaderos Hortduino está pensado para su instalación en un invernadero de reducidas dimensiones. Dadas las funciones de Hortduino, es necesaria una infraestructura de red que soporte la comunicación entre los diferentes subsistemas que componen este proyecto, ya sea mediante el uso de Internet o bien empleando una red Ethernet de área local, para el correcto funcionamiento del sistema.

Ilustración 12 – Despliegue del sistema Hortduino

Como se puede apreciar en la ilustración anterior, los subsistemas que componen Hortduino se pueden encontrar en la misma red o en diferentes redes, siempre que se puedan intercomunicar entre sí. Por cuestiones prácticas, en el desarrollo de este proyecto se ha optado por emplear un mismo equipo para albergar los subsistemas *Aplicación de Escritorio* y *Servidor XAMPP*. El dispositivo Arduino también se conecta en red local para realizar las pruebas pertinentes.

Los sensores encargados de medir las condiciones ambientales son instalados en lugares estratégicos del invernadero y su entorno de acuerdo a su cometido específico, como se explica en el apartado **3.2.2**.

6 PLANIFICACIÓN Y PRESUPUESTO

En la fase de planificación del proyecto, resulta de gran utilidad conocer el tamaño aproximado en líneas de código que ocupará el sistema, ya que dicho tamaño es un factor es clave a la hora de estimar el tiempo y el coste total del proyecto.

Para ello se empleará una metodología que estima este parámetro en función de los elementos lógicos que se utilizan en el sistema y de valoraciones subjetivas de su complejidad. Esta metodología se conoce por **Estimación por Puntos de Función** (Function Point Analysis) y fue definida por primera vez en el año 1979 por Allan J. Albrecht en el simposio de IBM sobre desarrollo de aplicaciones.

La métrica del punto función asigna una cantidad de puntos al software según la complejidad de los datos que maneja y los procesos que son realizados sobre esos datos, todo ello desde el punto de vista de la funcionalidad ofrecida al usuario. Los parámetros sobre los cuales se trabaja son:

- Número de entradas de usuario : Se cuenta cada entrada de usuario que proporciona diferentes datos empleados por la aplicación, como por ejemplo la alteración de los rangos aceptables de las variables controladas.
- Número de salidas de usuario : Son las salidas que proporcionan al usuario información sobre la aplicación en forma de pantallas, mensajes, informes, etc. Los elementos de datos particulares dentro de un informe no se cuentan de forma separada.
- Número de peticiones de usuario : Las peticiones del usuario son aquellas entradas que producen la generación de respuestas automáticas del software en forma de salida interactiva.
- Número de archivos : Cada uno de los archivos lógicos que forman parte del sistema, englobando en este concepto a las bases de datos.
- Número de interfaces externas : Se consideran todas las interfaces usables por el sistema que se utilizan para transmitir información a otro sistema.

Una vez que se han recopilado todos los elementos referentes a los puntos anteriores, se asocia a cada uno un factor de complejidad (alta, media o baja) en función del criterio adoptado. Este criterio es un tanto subjetivo, ya que está abierto a interpretaciones personales sobre lo que se considera complejo o sencillo. A continuación, se obtienen los Puntos de Función No Ajustados como la suma resultante de rellenar la *Tabla 1* que asocia cada parámetro con su factor de complejidad.

Para obtener los Puntos de Función ajustados se emplea la siguiente relación:

$$PF = PFNA * (0{,}65 + 0{,}01 * \sum FC_i)$$

siendo FC_i los valores de ajuste de complejidad asignados a los Factores de Valor de Ajuste Técnico (FVAT) en una escala de significación de 0 a 5, representando 0 la ausencia de influencia y 5 una influencia muy fuerte.

Cálculo de los Puntos de Función No Ajustados

	Factor de ponderación						
Parámetros de medición	**Simple**		**Medio**		**Complejo**		**Suma**
Número de entradas de usuario		x3		x4		x6	
Número de salidas de usuario		x4		x5		x7	
Número de peticiones de usuario		x3		x5		x6	
Número de archivos		x7		x10		x15	
Número de interfaces externas		x5		x7		x10	
					Cuenta Total PFNA		

Tabla 1 – Obtención de los Puntos de Función No Ajustados

Factores de Valor de Ajuste Técnico (FVAT)

1. Comunicación de datos
2. Proceso distribuido de datos
3. Rendimiento
4. Configuración
5. Volumen de transacciones
6. Entrada de datos en línea
7. Diseño para la eficiencia del usuario final
8. Actualización de datos en línea
9. Complejidad del procesamiento
10. Reusabilidad del código
11. Facilidad de instalación
12. Facilidad de operación
13. Instalación múltiple
14. Facilidad de cambio
15. Requerimientos de otras aplicaciones

6.1 ESTIMACIÓN DEL TAMAÑO

Una vez se ha explicado el método a seguir para estimar los Puntos de Función, se aplicará paso a paso lo visto anteriormente a este proyecto. Aunque el sistema Hortduino se compone de varios subsistemas diferenciados, se considera el conjunto total de todos ellos para hacer el cálculo final de los Puntos de Función.

- **Aplicación de Escritorio**
  - Entradas
    - Dirección IP y puerto para conexión con Arduino (en caso de error) *(baja)*
  - Salidas
    - Pantalla de información principal, la cual incluye información de los sensores en tiempo real, gráficos interactivos y tiempo de su última actualización, nivel de agua en depósito, estado de los sistemas de riego/ventilación, mensajes y alertas del estado de las variables *(alta)*
    - Pantalla de configuración de red para conexión con Arduino *(baja)*
  - Peticiones de usuario
    - Activar/desactivar sistemas de riego y ventilación *(baja)*
    - Modificar los rangos aceptables de las variables *(baja)*
    - Información sobre la posición del puntero del ratón en las gráficas *(baja)*
    - Cambiar el intervalo de actualización de las gráficas interactivas *(baja)*
  - Archivos
    - Base de datos de lecturas de los sensores *(media)*
    - Archivo de configuración de rangos de las variables y estados *(baja)*
    - Archivo de estado actual de las condiciones *(baja)*
  - Interfaces externas
    - Comunicación por red mediante protocolo UDP *(baja)*
    - Comunicación con servicio MySQL remoto *(baja)*

Aplicación Android

- Entradas
  - Selección de menú de navegación *(baja)*

- Salidas
  - Pantalla de información principal, la cual incluye información de los sensores en tiempo real, estado de los sistemas de riego/ventilación, mensajes y alertas del estado de las variables *(alta)*
  - Nivel de agua en depósito *(baja)*
  - Información de conexión *(baja)*
  - Menú de navegación *(baja)*

- Peticiones de usuario
  - Activar/desactivar sistemas de riego y ventilación *(media)*
  - Modificar los rangos aceptables de las variables *(alta)*
  - Mostrar menú de opciones *(baja)*

- Archivos
  - Archivo de configuración IP *(baja)*

- Interfaces externas
  - Comunicación por red mediante protocolo UDP *(baja)*

- **Página Web**

  - Entradas
    - Selección de intervalo en Últimas Lecturas *(baja)*
    - Selección de intervalo y variables en Estadísticas *(baja)*
    - Selección de intervalo en Temperaturas *(baja)*

  - Salidas
    - Sección Últimas Lecturas *(media)*
    - Sección Estadísticas *(media)*
    - Sección Estado Actual *(media)*
    - Sección Consumo de Agua *(baja)*
    - Sección Temperaturas *(baja)*
    - Sección Sobre Hortduino *(baja)*

  - Peticiones de usuario
    - Cambiar la página visualizada *(baja)*
    - Selección de las variables a mostrar en Estadísticas *(media)*

  - Archivos
    - Archivo de estado actual de condiciones *(baja)*
    - Archivo de hoja de estilos en cascada CSS *(baja)*

  - Interfaces externas
    - Comunicación con servicio MySQL remoto *(baja)*

Una vez determinados los parámetros y su complejidad, se calculan los Puntos de Función No Ajustados empleando la *Tabla* 1 para cada subsistema:

Aplicación de Escritorio

Parámetros de medición	Factor de ponderación						Suma
	Simple		Medio		Complejo		
Número de entradas	1	x3	0	x4	0	x6	3
Número de salidas	1	x4	0	x5	1	x7	11
Número de peticiones	4	x3	0	x4	0	x6	12
Número de archivos	2	x7	1	x10	0	x15	24
Número de interfaces externas	2	x5	0	x7	0	x10	10
					Cuenta Total PFNA		60

Tabla 2 – Cálculo de los PFNA de la Aplicación de Escritorio

Aplicación Android

Parámetros de medición	Factor de ponderación						Suma
	Simple		Medio		Complejo		
Número de entradas	1	x3	0	x4	0	x6	3
Número de salidas	3	x4	0	x5	1	x7	19
Número de peticiones	1	x3	1	x4	1	x6	13
Número de archivos	1	x7	0	x10	0	x15	7
Número de interfaces externas	1	x5	0	x7	0	x10	5
					Cuenta Total PFNA		47

Tabla 3 – Cálculo de los PFNA de la Aplicación Android

Página Web

Parámetros de medición	Factor de ponderación						Suma
	Simple		Medio		Complejo		
Número de entradas	3	x3	0	x4	0	x6	9
Número de salidas	3	x4	3	x5	0	x7	27
Número de peticiones	1	x3	1	x4	0	x6	7
Número de archivos	2	x7	0	x10	0	x15	14
Número de interfaces externas	1	x5	0	x7	0	x10	5
					Cuenta Total PFNA		62

Tabla 4 - Cálculo de los PFNA de la Página Web

Seguidamente, se pasa a calcular la suma de los valores de ajuste de complejidad asignados a cada Factor de Valor de Ajuste Técnico, según se ha considerado para cada subsistema:

Factores de Valor de Ajuste Técnico	Aplicación de Escritorio	Aplicación Android	Página Web
Comunicación de datos	5	5	5
Proceso distribuido de datos	1	1	0
Rendimiento	0	0	0
Configuración	1	0	0
Volumen de transacciones	1	1	1
Entrada de datos en línea	4	4	4
Diseño para la eficiencia del usuario	4	4	3
Actualización de datos en línea	3	2	2
Complejidad del procesamiento	3	2	1
Reusabilidad de código	1	0	1
Facilidad de instalación	1	1	0
Facilidad de operación	0	0	0
Instalación múltiple	0	0	0
Facilidad de cambio	0	0	0
Requerimiento de otras aplicaciones	1	1	1
Total	**25**	**21**	**18**

Tabla 5 – Asociación de factores de complejidad por aplicación

Los Factores de Ajuste Técnico obtenidos en la tabla anterior son usados para calcular los Puntos de Función ajustados mediante la relación vista anteriormente:

- Aplicación de Escritorio: $PF = 60 * (0{,}65 + 0{,}01 * 25) = 54$
- Aplicación Android : $PF = 47 * (0{,}65 + 0{,}01 * 21) = 40{,}42$
- Página Web : $PF = 62 * (0{,}65 + 0{,}01 * 18) = 51{,}46$

Con los datos anteriores, se puede aproximar el tamaño del software resultante en líneas de código mediante la relación entre el lenguaje de programación empleado en el desarrollo y los puntos de función obtenidos. Esta relación está comprendida en una tabla construida por Capers Jones, especialista en metodologías de Ingeniería del Software, donde aparece el valor por el que se deben multiplicar los puntos de función según el lenguaje usado.

El lenguaje *Processing* es similar en estructura y sintaxis a *Java*, por lo que emplearemos dicha relación en el cálculo del tamaño de la Aplicación de Escritorio. Se procederá de forma similar en el caso de la Aplicación Android, ya que se usa el mismo lenguaje que en la Aplicación de Escritorio. Para la página web se utiliza la equivalencia referida al lenguaje HTML en su versión 3.0.

Lenguaje	Nivel	Líneas de código por punto de función
Java	6	53
HTML 3.0	22	15

Tabla 6 – Extracto de la tabla[1] de Capers Jones

En la tabla de Capers Jones los lenguajes de mayor nivel se asocian con un número menor de líneas de código por punto de función. En este caso, los valores obtenidos de la tabla son 53 para *Processing* y 15 para HTML 3.0, lo que resulta en los siguientes tamaños estimados para las distintas aplicaciones:

- Aplicación de Escritorio : 53 * 54 = 2862 líneas de código

- Aplicación Android : 53 * 40,42 = 2142 líneas de código

- Página Web : 15 * 51,76 = 776 líneas de código

6.2 ESTIMACIÓN DEL COSTE DE DESARROLLO

Obtenido el tamaño relativo de cada aplicación o herramienta que forma parte del proyecto, es razonable estimar el esfuerzo necesario para completar cada uno de estos elementos así como el tiempo necesario hasta su finalización. Para ello se hará uso de un modelo matemático que emplea la estimación del tamaño del software, medido en líneas de código (LOC), y una serie de atributos dependientes de la clasificación del proyecto en uno de los tres grupos planteados por este método.

Este modelo de estimación de costes en software, denominado **COCOMO** (Constructive Cost Model), fue desarrollado por Barry W. Bohem en el año 1981 y define tres modos que representan el tipo de proyecto, siendo:

- Modo Orgánico: pequeño grupo de programadores trabajando en un proyecto de tamaño pequeño o medio con requerimientos medianamente flexibles.
- Modo Semilibre: grupo mediano de programadores que incluye a personas de diferente experiencia con requisitos menos flexibles que en el modo orgánico.
- Modo Empotrado: combinación de los anteriores modos pero con requisitos estrictos.

1 Tabla completa disponible en http://www.cs.bsu.edu/homepages/dmz/cs697/langtbl.htm

Las fórmulas básicas que se emplean para el cálculo del esfuerzo, tiempo de desarrollo y personal requerido son las siguientes:

- Esfuerzo requerido (personas/mes) : $E = a(Kl)^b * m(X)$

- Tiempo de desarrollo (meses) : $Tdev = c(E)^d$

- Personal necesario (desarrolladores) : $P = E / Tdev$

donde:

a,b,c,d son constantes definidas en la *Tabla 8*, correspondientes al modo de desarrollo.

Kl es el número de líneas de código, expresado en miles.

m(X) es el multiplicador que depende de 15 atributos considerados.

Para el cálculo del multiplicador *m(X)* se elige un valor asociado a cada factor considerado en el desarrollo de acuerdo a la tabla mostrada a continuación. Hay que destacar que esta elección tiene carga subjetiva y está sujeta a la percepción del desarrollador o a la experiencia previa en el desarrollo de proyectos similares.

Atributos	**Valor**					
	Muy Bajo	Bajo	Normal	Alto	Muy Alto	Extra
Fiabilidad	0,75	0,88	1,00	1,15	1,40	
Tamaño de la base de datos		0,94	1,00	1,08	1,16	
Complejidad del producto	0,70	0,85	1,00	1,15	1,30	1,65
Restricciones en tiempo de ejecución			1,00	1,11	1,30	1,66
Restricciones de memoria virtual			1,00	1,06	1,21	1,56
Volatilidad de la máquina virtual		0,87	1,00	1,15	1,30	
Tiempo de respuesta		0,87	1,00	1,07	1,15	
Capacidad del analista	1,46	1,19	1,00	0,86	0,71	
Experiencia en la aplicación	1,29	1,13	1,00	0,91	0,82	
Calidad de los programadores	1,42	1,17	1,00	0,86	0,70	
Experiencia en la máquina virtual	1,21	1,10	1,00	0,90		
Experiencia en el lenguaje de programación	1,14	1,07	1,00	0,95		
Técnicas actualizadas de programación	1,24	1,10	1,00	0,91	0,82	
Uso de herramientas software	1,24	1,10	1,00	0,91	0,83	
Limitaciones de tiempo de desarrollo	1,22	1,08	1,00	1,04	1,10	

Tabla 7 – Asignación de valor a los atributos relativos al esfuerzo

En este proyecto en cuestión, al ser de tamaño relativamente pequeño y disponer de un solo desarrollador, se ha optado por el modo orgánico en el modelo intermedio de COCOMO. Los valores de las constantes que aparecen en las fórmulas mostradas anteriormente son extraídos de la tabla siguiente:

Modo	a	b	c	d
Orgánico	3,20	1,05	2,50	0,38
Semilibre	3,00	1,12	2,50	0,35
Empotrado	2,80	1,20	2,50	0,32

Tabla 8 – Valor de las constantes según modo de desarrollo

Con la estimación del tamaño de cada componente de este proyecto realizada anteriormente mediante el análisis de puntos de función y los valores de las constantes referentes al esfuerzo, personal y tiempo de desarrollo obtenidas de la tabla anterior, la elección de los valores asociados a los atributos usados en el cálculo del multiplicador *m(X)* es el último paso antes de aplicar las fórmulas vistas anteriormente para obtener los valores estimados para cada subsistema.

Atributos	Subsistema		
	Aplicación de Escritorio	Aplicación Android	Página Web
Fiabilidad	0,75	0,75	0,75
Tamaño de la base de datos	0,94	0,94	0,94
Complejidad del producto	0,85	0,70	0,70
Restricciones en tiempo de ejecución	1,00	1,00	1,00
Restricciones de memoria virtual	1,00	1,00	1,00
Volatilidad de la máquina virtual	1,00	1,00	0,87
Tiempo de respuesta	0,87	0,87	1,00
Capacidad del analista	1,00	1,00	0,86
Experiencia en la aplicación	1,13	1,13	1,00
Calidad de los programadores	1,00	1,00	1,00
Experiencia en la máquina virtual	1,21	1,21	1,00
Experiencia en el lenguaje de programación	1,14	1,14	0,90
Técnicas actualizadas de programación	1,10	1,10	0,82
Uso de herramientas software	1,00	1,00	1,00
Limitaciones de tiempo de desarrollo	1,00	1,00	1,00
Multiplicador resultante	0,89	0,74	0,27
Esfuerzo (persona/mes)	8,62	5,24	0,66
Tiempo de desarrollo (mes)	5,67	4,69	2,14
Personal necesario	1,52	1,12	0,31

Tabla 9 – Atributos asignados para cada subsistema y previsiones

Analizando los datos obtenidos, se observa que el tiempo estimado total para el desarrollo del proyecto es de aproximadamente 12 meses y medio. Partiendo de la base de que el diseño y la funcionalidad de la aplicación Android replica las características principales de la aplicación de escritorio en dispositivos móviles, y que la implementación interna de dichas aplicaciones tiene bastantes puntos en común como consecuencia de la reutilización de código fuente, la estimación de tiempo en el desarrollo de la aplicación Android ha de ser reducido al menos en un 75% para ajustarse a la realidad. Este porcentaje se elije a partir de la proporción de funciones que son comunes a las dos aplicaciones consideradas.

Con ésta reducción del tiempo de desarrollo de la aplicación Android, el tiempo de desarrollo total del proyecto cambia a 5'67 + 1'17 + 2'14 ≈ 10 meses.

6.3 ESTIMACIÓN DE TAREAS

Tras concretar el diseño y los requisitos que ha de cumplir el sistema en las primeras fases del proyecto, se planea la ejecución de cada etapa dividiendo el trabajo a realizar en tareas diferenciadas y asignando a cada tarea un espacio temporal definido en función de varios factores. Cada tarea posee características tales como orden de ejecución, simultaneidad, prioridad o necesidad de que otra tarea se haya completado previamente. La planificación del proyecto engloba el orden de dichas tareas y su duración, haciendo posible un flujo de trabajo temporal.

A medida que se avanza en el desarrollo, pueden surgir problemas en la consecución de algunas tareas que pueden extenderlas en el tiempo mas allá de la previsión temporal inicial que se les asignó, afectando al resto y a la estimación temporal total del proyecto.

Durante el desarrollo de este proyecto se ha considerado una jornada de trabajo de 4 horas diarias debido a restricciones de tiempo impuestas al autor por cuestiones ajenas a él. Como se expuso anteriormente en el punto 2.2, la elaboración de la documentación se supedita a la terminación del proyecto para focalizar todo el esfuerzo en lo relativo a la implementación del software, lo cual implica que durante la realización del proyecto se generan esquemas y notas acerca de diseño, implementación y pruebas de los diferentes subsistemas para luego ser recopiladas e integradas en la documentación final del PFC.

La duración de este proyecto es de 175 días no consecutivos, con periodos de inactividad debidos a factores externos, como la espera por la entrega de componentes hardware necesarios en la construcción del prototipo hardware y cuestiones ajenas al proyecto.

En la siguientes ilustraciones se reflejan la consecución de las tareas, listadas bajo la descomposición de cada iteración, y su planificación en forma de diagrama de Gantt.

ID	Nombre	Fecha de inicio	Fecha de fin	Antecesores	Duración	Cost
192	Proyecto Hortduino	11/11/13	11/07/14		175	3.359,05
0	Fase de estudio	11/11/13	14/11/13		4	264
2	Estudio del problema	11/11/13	14/11/13		4	104
3	Estudio de la plataforma de hardware Arduino	11/11/13	14/11/13		4	104
7	Iteración I	15/11/13	2/12/13	0	12	204,5
10	Ensamblaje del prototipo hardware inicial	15/11/13	21/11/13	0	5	80
14	Instalación y configuración del entorno de desarrollo Arduino	22/11/13	26/11/13		3	46,5
16	Pruebas de conexión y recepción de datos desde Arduino	27/11/13	2/12/13	10,14	4	78
20	Iteración II	3/12/13	5/02/14	7	47	733
22	Análisis de requisitos de la Aplicación de Escritorio	3/12/13	11/12/13		7	98
25	Instalación y configuración del entorno de desarrollo Processing	12/12/13	16/12/13		3	46,5
27	Diseño conceptual y funcional de la interfaz de usuario	16/12/13	20/12/13		5	60
30	Diseño de la estructura de datos para las lecturas de sensores	23/12/13	25/12/13		3	36
33	Diseño del protocolo de comunicación con el prototipo hardware	26/12/13	1/01/14	30	5	60
38	Implementación de la Aplicación de Escritorio	2/01/14	22/01/14	22,25,27,30,33	15	232,5
52	Pruebas de la aplicación de escritorio	23/01/14	5/02/14	38	10	200
71	Iteración III	6/02/14	24/02/14	20	13	240,75
73	Diseño de la base de datos	6/02/14	12/02/14		5	60
77	Instalación y configuración del servidor XAMPP	10/02/14	12/02/14		3	41,25
75	Integración de la libreria Bezier.SQL en la Aplicación de Escritorio	12/02/14	17/02/14		4	62
82	Pruebas de interación base de datos / Aplicación de Escritorio	18/02/14	24/02/14	75,77	5	77,5
57	Iteración IV	24/02/14	20/03/14		19	355,5
224	Análisis de requisitos de la Aplicación Android	24/02/14	26/02/14		3	42
58	Diseño de la interfaz de usuario de la Aplicación Android	27/02/14	6/03/14	224	6	72
60	Implementación de la Aplicación Android	7/03/14	17/03/14	58	7	115,5
68	Pruebas de la Aplicación Android	10/03/14	20/03/14		9	126
87	Iteración V	21/03/14	28/04/14	71	27	490,3
88	Análisis de requisitos de la página Web	21/03/14	25/03/14		3	42
91	Diseño de la página Web	26/03/14	1/04/14	88	5	65
95	Diseño de los elementos gráficos y logos	26/03/14	31/03/14		4	52
206	Maquetación de la estructura de la página Web en archivo CSS	2/04/14	7/04/14	91,95	4	59,2
97	Diseño de los sketches de datos	2/04/14	8/04/14		5	60
104	Implementación de los sketches para la página Web	9/04/14	17/04/14	97	7	103,6
108	Pruebas de acceso y usabilidad	18/04/14	28/04/14	77,104	7	108,5
114	Documentación	11/11/13	11/07/14		175	1.071
115	Obtención de esquemas y notas para elaborar la documentación	11/11/13	24/04/14		119	595
117	Elaboración de la documentación final del proyecto	25/04/14	11/07/14	115	56	476

Ilustración 13 – Planificación de las tareas según las iteraciones propuestas

Se puede concluir a partir de las ilustraciones 13 y 14 que varias de las tareas relacionadas con las pruebas del software se superponen con algunas de las tareas de implementación de dicho software. Esto se debe al proceso continuo de mejora y corrección de errores que se ha usado durante el desarrollo de los diferentes subsistemas que componen Hortduino, permitiendo añadir nuevas funcionalidades al software a medida que son descubiertas nuevas necesidades en este proceso retroalimentado.

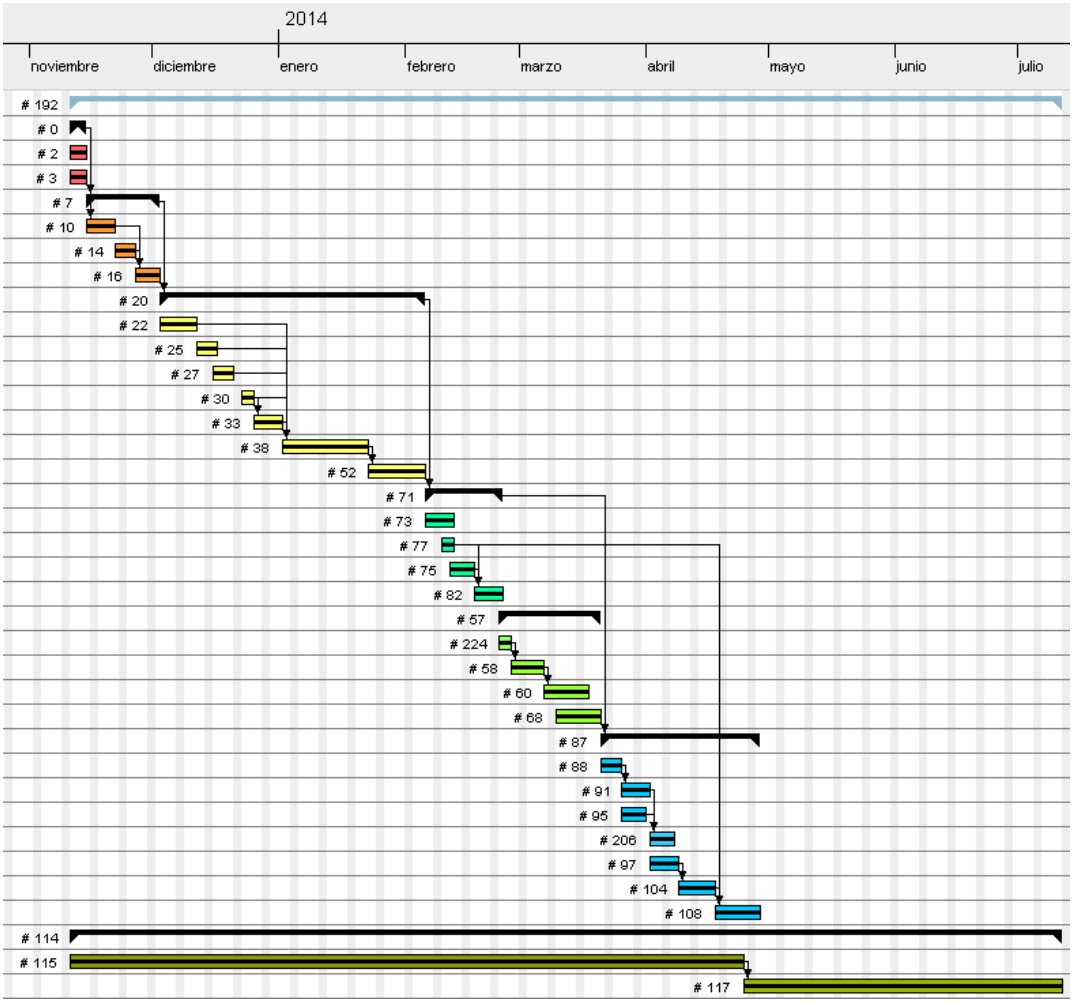


Ilustración 14 – Diagrama de Gantt

En el diagrama de Gantt de la ilustración 14 se observa el desarrollo en el tiempo de las diferentes tareas, con la numeración correspondiente a las indicadas en la ilustración 13. Las flechas indican que la tarea a la que apuntan necesita como requisito previo a su inicio la finalización de la tarea de la cual procede la flecha.

6.4 PRESUPUESTO DEL PROYECTO

Cada recurso disponible lleva asociado un coste y un grado de uso que refleja su utilización en la consecución de las tareas del proyecto en las que es empleado, por lo que el coste total del proyecto será la suma de los valores porcentuales de la utilización sobre el coste de cada recurso.

En la siguientes tablas se recopilan todos los recursos utilizados en la realización de este proyecto junto a su coste de adquisición y porcentaje de utilización. También se indican los recursos humanos con el coste del personal por hora de trabajo.

Recurso	Coste (€)	Utilización (%)	Coste asociado al proyecto (€)
Ordenador Personal	800,00	60	480
Smartphone Samsung Galaxy Mini II	110,00	25	27,5
Router TP-LINK TD-W8970	70,00	50	35
Placa Arduino Mega 2560	49,61	100	49,61
Shield Ethernet para Arduino	35,09	100	35,09
Sensor de temperatura y humedad DHT11	5,62	100	5,62
Sensor de temperatura National Semiconductor LM35	1,10	100	1,1
Sensor de distancia por ultrasonidos Seed U-S	18,15	100	18,15
Sensor de humedad del suelo	0,00	100	0
Fotorresistencia GL5528 LDR	1,45	100	1,45
Válvula de solenoide 12V	9,07	80	7,26
Ventilador 12V	3,00	80	2,4
Relés Grove-Twig 5V	7,26	80	5,81
Breadboard mini para prototipado	12,10	100	12,1
Set de jump cables breadboard	6,05	100	6,05
Resistencias eléctricas	0,40	100	0,4
Fuente de alimentación 9V	6,00	100	6
		TOTAL	693,53

Tabla 10 – Costes de los recursos hardware

Recurso	Coste (€)	Utilización (%)	Coste asociado al proyecto (€)
MS Windows 7 (incluido en el coste del PC)	0,00	100	0
Sublime Text 2 (trial version)	0,00	100	0
Processing 2.11 IDE	0,00	100	0
Arduino 1.0.5 IDE	0,00	100	0
Adobe Photoshop CS6 (trial version)	0,00	100	0
LibreOffice 4.2	0,00	100	0
GanttProject 2.7b	0,00	100	0
Fritzing 0.8.7b	0,00	40	0
XAMPP 1.8.3	0,00	100	0
Java JDK 1.7.0.45	0,00	80	0
yEd Graph Editor 3.12.2	0,00	100	0
Android SDK	0,00	80	0
		TOTAL	0 €

Tabla 11 – Costes de los recursos software

Recurso	Coste por hora de trabajo
Desarrollador / Analista / Encargado de pruebas	12 €
Documentador	5 €

Tabla 12 – Coste de los recursos humanos

Según lo indicado en la tabla 11, los costes totales asociados al software empleado en este proyecto se reducen a 0 €. Esto es debido a la utilización de programas y herramientas de acceso público gratuito o en su versión de prueba, lo cual reduce los costos empleados en licencias.

Con toda la información expuesta anteriormente, el coste total del proyecto resulta de la suma del coste de desarrollo indicado en la ilustración 13 y el coste de los recursos físicos empleados durante su realización, recogido en la suma total de la tabla 10:

3359,05 € + 693,53 € = **4052,58 €**

7 CUESTIONES DE DISEÑO E IMPLEMENTACIÓN RESEÑABLES

COMPONENTES DEL PROTOTIPO HARDWARE

El mercado de sensores electrónicos cuenta con un amplio abanico de opciones disponibles en lo que respecta a la medición de diferentes variables físicas. Para este proyecto se realizó un inventario preliminar de sensores basándose principalmente en la relación precisión/precio y en la documentación disponible sobre ellos. En el caso del sensor para medir la humedad del suelo, se optó por la fabricación propia.

En la construcción de este sensor se emplearon materiales asequibles y de uso común, como son el yeso y clavos galvanizados. Este sensor de la humedad del suelo es de tipo resistivo, ya que se basa en la conductividad a través de él en función de la humedad presente en el suelo que es absorbida por capilaridad hacia el cilindro de yeso.

Durante el periodo de pruebas se detectó corrosión en uno de los clavos debido a la reacción electrolítica que aparece al hacer circular corriente eléctrica entre los dos terminales. Para solucionar este problema, se optó por construir un nuevo sensor reemplazando los clavos galvanizados que actuaban como terminales en el sensor por dos barras de grafito, material mucho mas resistente a éste fenómeno pero que posee una resistividad propia superior a la del acero galvanizado, por lo que fue preciso efectuar la calibración de la medición a partir de los nuevos valores obtenidos reemplazando una de las resistencias del prototipo.

Para simular el comportamiento del nivel de agua en el depósito de riego, se empleó un potenciómetro capaz de variar su resistencia y ofrecer distintos valores en el pin de entrada asignado en la placa Arduino. Al necesitar una solución mas realista al problema de calcular el nivel de agua para riego, se optó por utilizar un sensor de distancia por ultrasonidos. Este sensor funciona por pulsos ultrasónicos que son emitidos perpendicularmente a la superficie del agua contenida en el depósito con el objetivo de medir el tiempo transcurrido entre el envío del pulso y el retorno del eco al rebotar en el agua. La ventaja de este sensor frente a otras soluciones propuestas es la adaptación a depósitos de diferente profundidad (hasta 4 metros) modificando ligeramente el código de la aplicación de escritorio, mostrado a continuación:

```
void lecturaPing () {

  unsigned long retorno = 0;          // Distancia calculada en cm.

  pinMode (pinEco, OUTPUT);
  digitalWrite (pinEco, LOW);
  delayMicroseconds (2);
  digitalWrite (pinEco, HIGH);         // Pulso -> )))
  delayMicroseconds (15);
  digitalWrite (pinEco, LOW);          // Fin del pulso
  delayMicroseconds(20);
  pinMode (pinEco, INPUT);             // Retorno del eco <- (((
  retorno = pulseIn(pinEco, HIGH);
  // Velocidad del sonido  29 cm/milisegundos (58 ida y vuelta)
  // La distancia real se ajusta entre el rango mínimo del sensor (3cm) y la
profundidad de depósito marcada por profDeposito. El valor que se envía es el
porcentaje de llenado del depósito de 0 a 100.
  nivelH2O = map(retorno/58,3,profDeposito,100,0);
```

```
  nivelH2O = nivelH2O * 10;              // Multiplicación para ajustarse al
                                            formato usado en Processing
  if (nivelH2O < 0) { nivelH2O=0; }      // Evitamos niveles negativos si el
                                            sensor lee fuera del rango
}
```

Cuadro 1 – Método de lectura del sensor ultrasónico

El periodo de lectura de las variables se ajustó a 1.5 segundos excepto en el caso de la humedad del suelo, donde se ha espaciado la lectura hasta los 10 segundos con el fin de no provocar un desgaste prematuro en el sensor y ahorrar energía. Hay que destacar que en el caso del sensor de temperatura externa, se detectaron fuertes variaciones en el valor ofrecido en la primera lectura efectuada en cada intervalo, por lo que se aplicó una corrección consistente en realizar 10 lecturas consecutivas con un pequeño retraso entre cada una y devolver la media del conjunto de datos para estabilizar las lecturas. El procedimiento de lectura de sensores y composición de los datos a enviar se detalla en el código fuente mas abajo:

```
void lecturaSensores (int pinLDR, int pinLM35, int pinYeso, byte pinDHT=DHTPIN,
byte tipoDHT=DHTTYPE, String datos ="") {

  unsigned long tiempo = millis();  // Tiempo transcurrido desde el inicio del loop
  char bufferTemp[5];               //Buffer para la conversión float/string
  float tempLM35 = 0.0;             // Temperatura exterior (en grados celsius)
  int humedadSuelo;
  int sumTemps = 0;

  int luminosidad = analogRead(pinLDR);
  int humedadDHT = dht.readHumidity();
  int tempDHT = dht.readTemperature();
  lecturaPing ();

  if (isnan (humedadDHT) || isnan (tempDHT)) {    // Comprobación de que la lectura
                                                     del sensor DHT11 es correcta
    Serial.println ("-- Error de lectura en DHT11 --");
    humedadDHT = 0;
    tempDHT = 0;
  }

if (tiempo - actHumSuelo > intervaloHumSuelo) {    // Lectura espaciada de la
humedad del suelo para evitar electrólisis en los materiales
    actHumSuelo = millis();
    digitalWrite (pin5V, HIGH);
    delay(5);
    humedadSuelo = 1023 - analogRead(pinYeso);    // Se resta la resistencia
                                                     ofrecida por el sensor de yeso
    humedadSueloAnt = humedadSuelo;               // Conservar la lectura
    digitalWrite (pin5V, LOW);
  }

  else {
    humedadSuelo = humedadSueloAnt;      // Mantenemos el valor de la anterior
                                            lectura en la variable humedadSuelo
  }

  tempLM35 = analogRead(pinLM35);        // Descartamos la primera lectura por alta
                                            fluctuación
```

```
  for (int i=0; i<=9; i++) {              // Obtiene 10 muestras de temperatura para
                                             mayor precisión
     sumTemps += analogRead (pinLM35);    // Las muestras son almacenadas en la
                                             variable sumTemps
     delay (100);                         // Retardo para estabilizar las lecturas
  }
// Correción de muestreo de temperatura según datasheet LM35 (10mV / °C) y voltaje
de referencia de 5V --> ((5*100*lectura)/1024)/10 muestras = 0.048875855
  tempLM35 = sumTemps * 0.048875855;

  datos += luminosidad;                   // Construye un string con todos los datos
                                             de los sensores separados por comas
  datos += ',';
  dtostrf (tempLM35,2,1,bufferTemp);      // Convierte el valor float con un decimal
                                             a string
  datos += String(bufferTemp);
  datos +=',';
  datos += tempDHT;
  datos += ',';
  datos += humedadDHT;
  datos += ',';
  datos += humedadSuelo;
  datos += ',';
  datos += nivelH2O;
  datosSensores = datos;                   // Contiene los datos formateados para
                                              enviar a Processing
  Serial.println (datosSensores);         // Debug serial
}
```

Cuadro 2 – Procedimiento de lectura de sensores en Arduino

En la ilustración 15 se representa el modelado del prototipo hardware inicial, realizado en el programa de diseño de prototipos *Fritzing*. Este programa permite exportar los esquemas del diseño a un formato compatible con la producción física del PCB, incluyendo los componentes necesarios y el ruteado de las pistas sobre la placa.

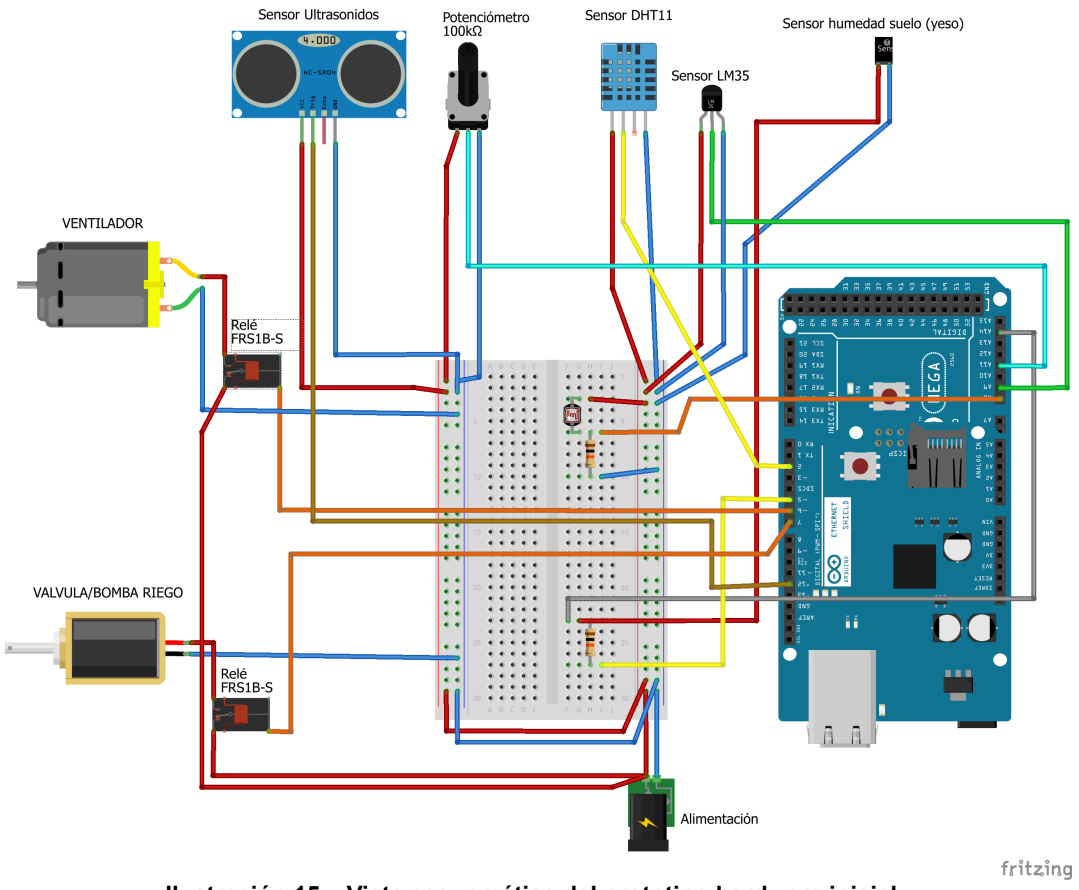


Ilustración 15 – Vista esquemática del prototipo hardware inicial

ESTRUCTURA DE DATOS Y PROTOCOLO

La transmisión de datos entre Arduino y el PC o dispositivo Android del usuario emplea un formato de datos muy sencillo basado en valores numéricos ordenados y separados por comas, todo ello es agrupado en una variable de tipo *string* que es enviada al dispositivo de destino.

Luminosidad	Temperatura Exterior	Temperatura Interior	Humedad Ambiental	Humedad Suelo	Nivel Agua

Para el envío de órdenes, se ha adoptado un sistema basado en tres dígitos que codifican la orden a ejecutar en el dispositivo Arduino de la siguiente forma:

Selector Sistema	Orden riego	Orden Ventilación

El primer campo siempre contiene un **1** para señalizar una nueva orden a los sistemas de riego y ventilación. El valor **0** está reservado para otro tipo de órdenes a sistemas que se implementen en una posible ampliación. El segundo campo contiene un **1** si la activación del sistema de riego es necesaria y un **0** en caso contrario. De forma similar, la última posición de las órdenes indica con un **1** la activación del sistema de ventilación o la desactivación del mismo si toma el valor **0**.

La aplicación de escritorio tiene la capacidad de reenviar la información recibida si registra un mensaje procedente de una IP distinta a la especificada para el dispositivo Arduino. Esto permite a la aplicación Android mostrar los mismos datos que maneja la aplicación de escritorio. Para mantener la sincronización entre las dos aplicaciones en lo relativo a la configuración actual del sistema, se modifica el sistema de envío de órdenes de la siguiente manera:

- Si la aplicación Android cambia la configuración, envía un mensaje a la aplicación de escritorio utilizando el campo [*SelectorSistema*] con el valor **1** para notificar a la aplicación de escritorio que debe guardar la nueva configuración indicada en los campos siguientes del mensaje en el archivo de configuración del servidor.
- Si la aplicación de escritorio realiza un cambio en la configuración mientras la aplicación Android está conectada, se sustituye el mensaje conteniendo los datos recibidos desde Arduino por un mensaje que solo contiene el carácter '*c'* para indicar que la aplicación de Android debe recargar la configuración desde el archivo almacenado en el servidor.

Dada la naturaleza de trabajo casi en tiempo real del sistema, es necesario que el envío de los datos recabados por los sensores y la generación de las respuestas del sistema en forma de órdenes sean efectuadas de forma rápida para adaptarse a un cambio de condiciones. A esto se suma la necesidad de que, ante un fallo en la comunicación entre Arduino y el dispositivo del usuario, los datos recibidos mas recientemente tengan prioridad sobre aquellos que no llegaron a tiempo al destino, descartándose estos últimos. Por estos motivos fue elegido el protocolo de comunicación en red UDP frente al protocolo TCP, puesto que proporciona las características requeridas para el funcionamiento del sistema y supone un ahorro en el volumen de datos a transmitir al prescindir de las funcionalidades asociadas al protocolo TCP como son el establecimiento previo de la conexión y la confirmación de llegada de cada paquete enviado.

El método perteneciente al handler de escucha UDP *receive()* de la librería *hypermedia.net* para *Processing* puede ampliar su funcionalidad incorporando mas instrucciones añadidas por el desarrollador que ejecuten tareas auxiliares cada vez que hay un evento de recepción de datagramas en el puerto de escucha. A continuación se muestra el código que emplea este handler en la aplicación de escritorio:

```
void receive (byte[] mensaje, String ipAdress, int port) {    // Handler por
defecto de la libreria UDP que se activa en el evento de recepción de datagramas

  if  ( ipAdress.equals(ip) )        // La dirección IP que manda el mensaje
                                        es la del Arduino
    if (mensaje.length > 11) {
      nuevosDatos = true;
      conexion = true;               // Si recibe un paquete UDP marca la conexión
                                        como activa
      paquetesEsperados = 0;         // y pone a cero el contador de
                                        paquetesEsperados
      datos = new String(mensaje);
      arrayDatos = splitTokens (datos,",");          // Trocea la cadena de entrada
a un array, usando una coma como separador de valores
```

```
    }
    else {
      println ("Array datos incompleto!");
      for (int i=0; i<6; i++) { arrayDatos[i] = "0"; }
      nuevosDatos = false;
    }

  else {   // La dirección IP que se conecta datos pertenece al dispositivo Android

    ipExterna = ipAdress;
    udp.send (datos, ipAdress, port);        // Reenvia los datos al dispositivo
                                                Android conectado

    if (char(mensaje[0])=='1')  {            // Se han recibido ordenes desde
                                                Android

      switch (mensaje[1]) {
        case 48:
          estadoOrdenes[0] = 0;
          break;
        case 49:
          estadoOrdenes[0] = 1;
          break;
        case 50:
          estadoOrdenes[0] = 2;
        break;
      }

      switch (mensaje[2]) {
        case 48:
          estadoOrdenes[1] = 0;
          break;
        case 49:
          estadoOrdenes[1] = 1;
          break;
        case 50:
          estadoOrdenes[1] = 2;
        break;
      }
    salvarConfig();               // El mensaje recibido contiene ordenes nuevas y
se necesita guardar la configuración y refrescar las condiciones
    recargarCondiciones = true;
    }
  }
}
```

Cuadro 3 – Método *receive* empleado en la aplicación de escritorio

DISEÑO DE LA BASE DE DATOS

La información recibida desde el dispositivo Arduino resulta de gran utilidad a largo plazo si se quiere analizar la evolución de los cultivos y calificar el resultado final de la cosecha en relación a las condiciones establecidas para su crecimiento. Para este fin, se dispuso de una única tabla que almacena los registros con la siguiente estructura:

Columna	Tipo	Nulo	Predeterminado
Indice	int(11)	No	
Fecha	timestamp	No	CURRENT_TIMESTAMP
Luminosidad	int(4)	No	
TempExterior	float	No	
TempInterior	int(3)	No	
HumAmbiente	int(3)	No	
HumSuelo	int(4)	No	
NivelAgua	int(4)	No	
EstadoRiego	int(1)	No	
EstadoVent	int(1)	No	

Tabla 13 – Estructura interna de la base de datos

La clave primaria es **Indice** y se utiliza para indexar cada nuevo registro de forma unívoca, siendo autoincrementada al añadir nuevas entradas. A parte de las columnas relativas a la fecha de la lectura y los valores registrados por los sensores, se incluyen también el nivel de agua y los estados del riego y la ventilación en el momento de la lectura. Esto permite efectuar el cálculo del volumen de agua utilizado en un determinado periodo temporal y comprobar en qué momentos y durante cuánto tiempo han estado en funcionamiento el riego y la ventilación del invernadero.

INFORMACIÓN VISUAL EN LA PÁGINA WEB

Con objeto de disponer de la información relativa a las condiciones actuales y anteriores del invernadero en cualquier lugar, la página web del proyecto Hortduino se diseñó para generar contenido útil para el usuario de forma dinámica a partir de la información disponible almacenada en la base de datos del servidor. Para este fin se idearon una serie de elementos que muestran los datos obtenidos por el sistema de forma gráfica e intuitiva para el usuario, el cual puede visualizar diversos parámetros relacionados entre sí, como por ejemplo la evolución temporal del valor de la humedad presente en el suelo frente al nivel restante de agua en el depósito y los periodos en los que el sistema de riego ha estado activo.

Aprovechando las oportunidades ofrecidas por el lenguaje *Processing* y su integración en la web usando el lenguaje HTML 5, se optó por la combinación de ambos en el diseño de los *sketches* que muestran la información al usuario en detrimento de otros métodos de generación de gráficos basados en el empleo de librerías gráficas predefinidas de terceros.

Ante el problema de representar comparativamente en un mismo gráfico variables que usan diferentes rangos y unidades de medida, se recurrió a la agrupación de los parámetros que utilizan las mismas unidades de medida en las partes derecha e izquierda de la gráfica, correspondientes al eje Y, en la sección *Estadísticas* de la página web.

Las variables pertenecientes a cada grupo están diferenciadas por su ubicación en la leyenda del gráfico, lo que hace sencillo su distinción. Para mayor claridad, la numeración de la escala en el eje Y correspondiente solo aparece si se ha seleccionado la variable a mostrar en la leyenda del gráfico, ahorrando información innecesaria que pueda llevar a confusión.

CONFIGURACIONES Y ESTADOS

En lo referente a la información relativa a las condiciones actuales del invernadero, se emplean varios ficheros de texto plano que sirven como elemento sincronizador entre los diferentes subsistemas de Hortduino. Estos ficheros son actualizados por la aplicación de Escritorio y la aplicación para dispositivos Android (a través de la anterior) con el objeto de hacer persistente la configuración actual de rangos aceptables y el estado de los sistemas de riego y ventilación. Al estar compuestos por texto plano, es posible simular diversos escenarios útiles a la hora de ejecutar las pruebas de cada sistema.

LENGUAJES EMPLEADOS

La elección del lenguaje *Processing* para este proyecto se ha justificado anteriormente en el punto 2.3 de la la sección *TECNOLOGIAS DE DESARROLLO*, basándose principalmente en la buena integración con Arduino y las posibilidades de portar la aplicación al entorno Android. Desgraciadamente, este lenguaje no proporciona funciones en su API para guardar información directamente en un archivo remoto, lo cual es necesario para el funcionamiento del sistema Hortduino, dado su carácter distribuido. La posibilidad de leer la información contenida en un archivo alojado en otra máquina si que está disponible en la API de *Processing*, por lo que es la única posibilidad de trabajar con ficheros externos.

Para subsanar este problema, se implementó un método[2] basado en la invocación remota de un script PHP que es capaz de ejecutar las funciones de carga/guardado de la configuración elegida por el usuario sobre los valores mínimos y máximos de las variables y actualizar el estado de las condiciones actuales, todo ello dependiendo de los argumentos suministrados a dicho script. De esta manera, sólo hay que recurrir a la función de *Processing* que efectúa la lectura de información de un archivo remoto para que el intérprete PHP ejecute la acción requerida remotamente con los parámetros suministrados y genere el resultado en el archivo remoto objeto de la modificación solicitada.

APLICACIÓN PARA DISPOSITIVOS ANDROID

Esta aplicación se idea como una extensión de la aplicación principal de escritorio tras descubrir la fácil adaptación de las aplicaciones construidas con *Processing* al entorno Android. Los únicos problemas encontrados son los referentes a la interfaz de usuario, ya que no se pueden

2 Idea extraída de http://www.learningprocessing.com/

utilizar los componentes disponibles en la GUI de Android para mostrar botones o elementos gráficos interactivos al ser el método de dibujado en pantalla incompatible con *Processing*, por lo que se ha implementado su funcionalidad y comportamiento en clases específicas para esta aplicación. El resultado obtenido no es del todo óptimo de cara al usuario, pero es aceptable en el desempeño de la aplicación.

8 CONCLUSIONES Y POSIBLES AMPLIACIONES

8.1 CONCLUSIONES

Plataforma de desarrollo Open-Hardware

La plataforma de hardware libre Arduino posee un gran potencial en proyectos relacionados con todo tipo de actividades que incluyan la adquisición de datos a través de sensores. Además de su bajo coste, Arduino cuenta con el respaldo de una gran comunidad mundial de aficionados que ofrecen multitud de ejemplos y tutoriales, algunos de los cuales han sido de gran ayuda en la elaboración de este proyecto. Otro factor que hace interesante a esta plataforma es que sigue la filosofía *OpenHardware*, similar a la conocida *OpenSource* en lo relacionado al software, que posibilita las modificación por parte de los usuarios de las especificaciones del hardware sin restricción alguna, adaptándose a multitud de configuraciones específicas que se requieran y favoreciendo la innovación en el diseño.

Internet of Things

El llamado *Internet of Things* es un concepto interesante sobre el que se ha profundizado al realizar este proyecto. En un mundo cada vez mas interconectado entre sí, las aplicaciones del *IoT* abarcan ámbitos tan diferentes como el control ambiental, la seguridad ciudadana, gestión energética eficiente, domótica, aplicaciones médicas, control de tráfico e infraestructuras entre otras muchas áreas en las que se puede incorporar. Aplicar este concepto a la vida diaria puede suponer una mejora en la calidad de vida y en el entorno que nos rodea, por ello, el *IoT* es un campo de trabajo con posibilidades de cara al futuro.

Entornos de desarrollo

Mientras se tomaba contacto con la plataforma Arduino y su lenguaje en lo relativo a encontrar formas de comunicación entre el dispositivo y el PC, el lenguaje *Processing* apareció recomendado en la referencia de Arduino como candidato a implementar esta tarea.

El entorno de desarrollo de *Processing* está basado en *Java*, lo cual proporciona al desarrollador una extensa documentación disponible, con tutoriales y ejemplos, que hacen el desarrollo de las tareas mas sencillo. Por su parte, el lenguaje Arduino usa como base *Wiring*, un framework para la programación de microcontroladores basado en C/C++. Ambos lenguajes

comparten muchos puntos en común, pero con algunas diferencias menores, lo que resultó en una mayor fluidez en las tareas de programación y testeo.

Hay que destacar que, aunque no se ha hecho uso de este recurso en el proyecto, existe un firmware específico para el microcontrolador llamado *Firmata* que proporciona acceso a las funciones propias del lenguaje Arduino desde *Processing* sin tener que programar previamente el microcontrolador, pero limitando la comunicación entre los dos dispositivos al puerto serial.

- **Interfaces simples y sencillas**

La facilidad de utilización de una aplicación depende directamente del diseño de la interfaz que ésta ofrece al usuario. El diseño de las interfaces de las aplicaciones escritorio y Android fue el resultado de la incorporación sucesiva de los diferentes elementos que las componen, enfatizando en la claridad y el aspecto visual sencillo e intuitivo.

Se decidió incluir imágenes animadas en la representación del estado de los sistemas de riego y ventilación para dar un mayor énfasis a esta información, ya que es de vital importancia que el usuario conozca estos datos a primera vista. Para la representación de las condiciones presentes en el invernadero también se han utilizado imágenes descriptivas que hacen mas fácil la identificación de las condiciones anómalas en el invernadero.

Todo ello forma un conjunto de información que es ofrecida de modo claro y directo al usuario con el uso de elementos visuales fácilmente reconocibles.

8.2 POSIBLES AMPLIACIONES Y MEJORAS

- **Sistemas de humidificación y regulación de luz**

Para completar el control sobre las variables del invernadero, estos dos nuevos sistemas podrían ser implementados en un futuro como reguladores del crecimiento de las cosechas. El sistema de humidificación se apoyaría en las lecturas del sensor de humedad interior para actuar sobre pulverizadores que aumentasen la cantidad de vapor de agua presente en el aire del interior del invernadero. Por otra parte, el sistema de regulación de luz controlaría varios servomotores encargados del cierre o apertura de las mallas de sombreado.

- **Registro de alertas y notificaciones vía Twitter**

Otra funcionalidad que se podría integrar en el sistema Hortduino es la capacidad de alertar al usuario vía Twitter cuando alguna de las condiciones del invernadero se encontrase en situación de alerta. Estas alertas se registrarían en una tabla de la base de datos para poder incorporar dicha información al histórico de actividad del invernadero si se quiere consultar mas adelante.

- **Control directo de los sistemas desde la Web**

Esta nueva funcionalidad añadiría la posibilidad de que un usuario, previamente autentificado en una conexión segura con la página web, accediera a los controles de los sistemas de riego, ventilación y pudiese cambiar los valores aceptables para las variables supervisadas.

- **Captura de imágenes interiores**

Mediante el uso de una cámara sencilla, el usuario podrá ver en todo momento el estado de crecimiento de los cultivos desde una ubicación lejana. Este sistema puede aportar funciones de seguridad y vigilancia remota, así como construir un *time-lapse* del desarrollo de la cosecha con las imágenes obtenidas a partir de la cámara en intervalos regulares a lo largo de días o meses.

- **Sistema autónomo de alimentación**

Para dotar al sistema Hortduino de una fuente de energía limpia, este sistema sería el encargado de almacenar la energía solar recibida durante el día en una batería para su uso nocturno. También contaría con un módulo propio en la interfaz de las aplicaciones con funciones de información de la carga actual de la batería, tasa de recarga, coste por tiempo de funcionamiento ahorrado al sistema y origen de la energía que se utiliza actualmente (red eléctrica o generación propia).

- **Almacenaje de datos en Arduino**

Aprovechando la presencia de un zócalo para la tarjeta de almacenamiento mini-SD en la placa Arduino, se podría usar esta posibilidad con el fin de guardar una copia de los datos enviados por red en un archivo con formato CSV (Comma Separated Values) para el posterior tratamiento de los datos recopilados o bien almacenar las imágenes que capture el sistema de supervisión, una vez implementado e instalado.

- **Adaptación de la interfaz de la aplicación Android a varias resoluciones**

El diseño de la interfaz actual de la aplicación está pensado para una resolución de pantalla de 640 x 480 pixeles, ya que es la disponible en el dispositivo utilizado para las pruebas. Para adaptar la interfaz a otras resoluciones serían necesarios elementos gráficos de varios tamaños, empleados en función de la información que el dispositivo ofrece sobre la pantalla, cambiar la disposición relativa del texto y usar el tamaño de fuente adecuado para cada caso.

- **Ampliar las opciones de configuración en la aplicación de escritorio**

Esta ampliación añadiría mas funcionalidades de configuración disponibles para el usuario en la ventana de la aplicación de escritorio acerca del tamaño y capacidad del depósito de riego, estado de la conexión con la aplicación Android, sonidos de alerta, etc...

9 BIBLIOGRAFÍA

- Arduino. *The Arduino playground wiki* [en línea] http://playground.arduino.cc/
- Arduino. *Arduino language reference* [en línea] http://arduino.cc/en/Reference/HomePage
- Processing. *Processing 2 language reference* [en línea] http://processing.org/reference/
- Stephane Cousot. *UDP library for Processing* [en línea] http://ubaa.net/shared/processing/udp/index.htm
- Daniel Shiffman. *Learning Processing* [en línea] http://www.learningprocessing.com/
- Processing.js team. *Processing.js Quick Start* [en línea] http://processingjs.org/articles/p5QuickStart.html
- Apache Friends. *XAMPP Installer*[en línea] https://www.apachefriends.org/index.html
- Refnes Data. *W3Schools Online CSS reference* [en línea] http://www.w3schools.com/cssref/default.asp
- The PHP Group. *PHP documentation and reference* [en línea] http://php.net/docs.php
- W3C Consortium. *HTML5 online reference* [en línea] http://dev.w3.org/html5/html-author/

10 ANEXO

GLOSARIO

Internet of Things (IoT)

El denominado 'Internet de las cosas' es un concepto que se refiere a la interconexión de objetos cotidianos con identificadores únicos, formando una red capaz de albergar, generar y transmitir datos entre los elementos que la conforman sin necesidad de actuación humana.

Data-relaying

Se denomina así a la retransmisión de datos que efectúa un intermediario entre el emisor y el receptor, sin alterar el contenido de dichos datos.

IP

Una dirección IP es una etiqueta numérica que identifica, de manera lógica y jerárquica, a una interfaz de un dispositivo dentro de una red que utilice el protocolo IP.

SQL

Son las siglas en inglés de *Structured Query Language* o lenguaje de consulta estructurado. Es un lenguaje declarativo de acceso a bases de datos relacionales que permite especificar diversos tipos de operaciones en ellas.

HTML

El lenguaje de marcas de hipertexto (*HyperText Markup Language)* es un lenguaje usado para codificar la estructura de un documento mediante marcas que contienen información adicional sobre la estructura del texto contenido en dicho documento. Es el lenguaje de marcado mas extendido, usado habitualmente para codificar páginas web.

CSS

Un archivo de hoja de estilos en cascada (*Cascade Style Sheet)* contiene el formato y aspecto de un documento escrito en un lenguaje de marcado, como HTML. Su principal función es la de separar el estilo de una página web de su contenido.

XAMPP

XAMPP es un entorno de desarrollo web compuesto por el servidor web Apache, el gestor de bases de datos MySQL y los intérpretes para lenguajes de script PHP y Perl. Su nombre viene de agrupar las iniciales de las herramientas que lo componen y añadir una X que representa a cualquiera de los sistemas operativos (*cross platform, X).* Es una herramienta de software libre.

PHP

Es un lenguaje de programación de scripts diseñado para el desarrollo de webs de contenido dinámico. El código PHP se ejecuta siempre en el servidor y el resultado es devuelto en forma de página web al usuario.

LICENCIA

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

BLOQUE II

DOCUMENTACIÓN TÉCNICA

Índice de contenido

Índice de ilustraciones

Índice de tablas

1 ANÁLISIS DE REQUISITOS DEL SISTEMA

1.1 INTRODUCCIÓN

El presente Proyecto Fin de Carrera está dedicado a desarrollar un sistema de monitorización y control de invernadero apoyado en la popular plataforma de hardware libre Arduino y en tecnologías web. Este sistema, compuesto por tres aplicaciones diferenciadas, recibirá datos acerca de las principales variables implicadas en el crecimiento de los cultivos, como pueden ser la humedad del suelo, humedad ambiental y temperatura, obtenidos mediante una red de sensores colocados en el invernadero.

La información enviada desde los sensores será procesada por la aplicación de escritorio, la cual se encargará de su almacenamiento en una base de datos externa y de la automatización y control del riego y ventilación mediante sendos actuadores mecánicos. Esta aplicación de escritorio también será capaz de mostrar los datos recibidos por red desde los sensores en tiempo real y establecer las condiciones máximas y mínimas que se han de mantener para cada variable, alertando al usuario si se encuentran fuera de rango. El usuario podrá seleccionar el modo automático o manual para el control del riego y la ventilación, de modo que se active el sistema de riego si la humedad del suelo cae por debajo de un determinado valor mínimo aceptable o se accione el sistema de ventilación si la temperatura o humedad interior superan un límite máximo.

Para apoyar a la aplicación de escritorio, la aplicación móvil será una réplica en versión reducida de las funcionalidades e interfaz de la aplicación de escritorio, contando con las ventajas de movilidad que ofrecen los dispositivos *ANDROID*. A diferencia de la aplicación de escritorio, la aplicación móvil no tendrá la funcionalidad de incorporar los datos recibidos a la base de datos externa, pero si podrá ajustar los rangos aceptables de las variables monitorizadas y activar/desactivar los controles de riego y ventilación.

Por último, se dispondrá de una página web donde el usuario podrá visualizar gráficamente la información almacenada en la base de datos por la aplicación principal y tener acceso a estadísticas y gráficos históricos de la evolución de las condiciones ambientales del cultivo durante diferentes periodos temporales. En dicha página web se mostrará de forma dinámica la última lectura recibida en los 30 segundos anteriores, el estado de las alertas y sistemas de riego y ventilación, el consumo de agua para riego y el porcentaje restante disponible en el depósito, entre otra información útil para el usuario.

A continuación se enumeran los objetivos y requisitos del sistema, tabulados de acuerdo a la plantilla recomendada en *Metodología para la Elicitación de Requisitos de Sistemas Software*[1] [A. Durán, B. Bernardez] (2000). Se ha escogido esta forma de presentación por su claridad y estandarización.

1 Documento disponible en *www.lsi.us.es/~informes/lsi-2000-10.pdf*

1.2 OBJETIVOS DEL SISTEMA

OBJ-01	**Recibir información de sensores remotos**
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Descripción	El sistema deberá recibir la información de forma remota desde diversos sensores
Importancia	Importante
Urgencia	Alta
Estado	Validado
Estabilidad	alta
Comentarios	Este objetivo es atribuible a las aplicaciones de escritorio y Android

Tabla 1 - Objetivo del sistema - Recibir información de sensores remotos

OBJ-02	**Mostrar información de sensores remotos**
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Descripción	El sistema deberá mostrar la información recibida de forma remota desde diversos sensores
Importancia	Importante
Urgencia	Alta
Estado	Validado
Estabilidad	alta
Comentarios	Este objetivo es atribuible a las aplicaciones de escritorio y Android. Incluye la representación numérica y gráfica de los datos recibidos.

Tabla 2 - Objetivo del sistema - Mostrar información de sensores remotos

OBJ-03	**Almacenar información recibida**
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Descripción	El sistema deberá ser capaz de almacenar los datos recibidos desde los sensores remotos en una base de datos
Importancia	Importante
Urgencia	Alta
Estado	Validado
Estabilidad	alta
Comentarios	Este objetivo es atribuible a la aplicación de escritorio

Tabla 3 - Objetivo del sistema - Almacenar la información recibida

OBJ-04	Cambiar el estado de los sistemas de riego/ventilación
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Descripción	El sistema deberá ser capaz de cambiar el estado de los sistemas de riego y ventilación del invernadero de forma remota a petición del usuario.
Importancia	Importante
Urgencia	Alta
Estado	Validado
Estabilidad	alta
Comentarios	Este objetivo es atribuible a las aplicaciones de escritorio y Android

Tabla 4 - Objetivo del sistema - Cambiar estado de los sistemas de riego/ventilación

OBJ-05	Mostrar el estado y modo de funcionamiento de los sistemas de riego/ventilación
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Descripción	El sistema deberá ser capaz de mostrar el estado de los sistemas de riego y ventilación del invernadero.
Importancia	Importante
Urgencia	Alta
Estado	Validado
Estabilidad	alta
Comentarios	Este objetivo es atribuible a las aplicaciones de escritorio y Android

Tabla 5 - Objetivo del sistema - Mostrar el estado y modo de funcionamiento de los sistemas de riego y ventilación

OBJ-06	Control automatizado de los sistemas de riego/ventilación
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Descripción	El sistema deberá ser capaz de cambiar el estado de los sistemas de riego y ventilación del invernadero de forma remota en base a los parámetros establecidos por el usuario
Importancia	Importante
Urgencia	Alta
Estado	Validado
Estabilidad	alta
Comentarios	Este objetivo es atribuible a las aplicaciones de escritorio y Android

Tabla 6 - Objetivo del sistema - Control automatizado de los sistemas de riego/ventilación

OBJ-07	**Consulta de información vía web**
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Descripción	El sistema deberá ser capaz de mostrar de forma gráfica información relevante para el usuario en la página web asociada. Esto contempla la última lectura recibida desde los sensores remotos, el estado de los sistemas de riego y ventilación, las temperaturas registradas, el nivel del depósito para riego y el histórico de la evolución de las variables monitorizadas.
Importancia	Importante
Urgencia	Normal
Estado	Validado
Estabilidad	alta
Comentarios	n/a

Tabla 7 - Objetivo del sistema - Consulta de información vía web

OBJ-08	**Informar sobre las condiciones actuales del invernadero**
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Descripción	El sistema deberá ser capaz de mostrar información acerca de las condiciones de temperatura, humedad y nivel de agua referentes al invernadero
Importancia	Importante
Urgencia	Alta
Estado	Validado
Estabilidad	alta
Comentarios	Este objetivo es atribuible a las aplicaciones de escritorio y Android y a la página web

Tabla 8 - Objetivo del sistema - Informar sobre las condiciones actuales del invernadero

1.3 REQUISITOS DE INFORMACIÓN

RI-01	**Información procedente de los sensores remotos**
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Objetivos asociados	OBJ-01 Recibir información de sensores remotos OBJ-03 Almacenar la información recibida
Requisitos asociados	RF-01 Recibir información sobre luminosidad RF-02 Recibir información sobre temperatura exterior RF-03 Recibir información sobre temperatura interior RF-04 Recibir información sobre humedad ambiental RF-05 Recibir información sobre humedad del suelo RF-06 Recibir información sobre nivel de agua RF-15 Almacenar lecturas y estados en la base de datos
Descripción	El sistema deberá almacenar y acceder a la información enviada por los sensores remotos
Datos específicos	• Valor de la luminosidad registrada en el sensor remoto • Valor de la temperatura exterior registrada en el sensor remoto • Valor de la temperatura interior registrada en el sensor remoto • Valor de la humedad ambiental registrada en el sensor remoto • Valor de la humedad del suelo registrada en el sensor remoto • Valor del nivel de agua registrada en el sensor remoto
Intervalo temporal	Indefinido
Importancia	Importante
Urgencia	Alta
Estado	Validado
Estabilidad	Alta
Comentarios	La información recibida sobre las lecturas de los sensores remotos es atribuible a la aplicaciones de escritorio

Tabla 9 - Requisitos de información - Información sobre los sensores remotos

RI-02	Información sobre el estado actual de los sistemas de riego y ventilación
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Objetivos asociados	OBJ-05 Mostrar el estado de los sistemas de riego/ventilación
Requisitos asociados	RF-13 Guardar estado de los sistemas de riego y ventilación
Descripción	El sistema deberá almacenar el estado de actividad de los sistemas de riego y ventilación
Datos específicos	• El sistema de ventilación puede estar en funcionamiento o no • El sistema de riego puede estar en funcionamiento o no
Intervalo temporal	Presente
Importancia	Importante
Urgencia	Alta
Estado	Validado
Estabilidad	Alta
Comentarios	La información referente al estado actual de los sistemas de riego y ventilación es atribuible a las aplicaciones de escritorio y Android y a la página web

Tabla 10 - Requisitos de Información - Información sobre el estado actual de los sistemas de riego y ventilación

RI-03	Información sobre el modo de control de los sistemas de riego y ventilación
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Objetivos asociados	OBJ-06 Control automatizado de los sistemas de riego/ventilación
Requisitos asociados	RF-13 Guardar estado de los sistemas de riego y ventilación
Descripción	El sistema deberá almacenar el modo de control de control de cada sistema
Datos específicos	• El sistema de ventilación puede estar en modo manual apagado (OFF), modo manual encendido (ON) o modo automático (AUTO) • El sistema de riego puede estar en modo manual apagado (OFF), modo manual encendido (ON) o modo automático (AUTO)
Intervalo temporal	Indefinido
Importancia	Importante
Urgencia	Alta
Estado	Validado
Estabilidad	Alta
Comentarios	La información referente al estado actual de los sistemas de riego y ventilación es atribuible a las aplicaciones de escritorio y Android y página web

Tabla 11 - Requisitos de Información - Información sobre el modo de control de los sistemas de riego y ventilación

RI-04	Información sobre la configuración de rangos aceptables de variables
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Objetivos asociados	OBJ-06 Control automatizado de los sistemas de riego/ventilación OBJ-08 Informar sobre las condicione actuales del invernadero
Requisitos asociados	RF-14 Guardar configuración de los rangos aceptables de las variables
Descripción	El sistema deberá almacenar la configuración del usuario acerca del valor máximo y mínimo aceptable para cada variable monitorizada
Datos específicos	• Valor máximo • Valor mínimo
Intervalo temporal	Indefinido
Importancia	Importante
Urgencia	Alta
Estado	Validado
Estabilidad	Alta
Comentarios	La información referente al estado actual de los sistemas de riego y ventilación es atribuible a las aplicaciones de escritorio y Android

Tabla 12 - Requisitos de Información - Información sobre la configuración de rangos aceptables en variables

RI-05	Información sobre el estado actual de las condiciones del invernadero
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Objetivos asociados	OBJ-08 Informar sobre las condiciones actuales del invernadero
Requisitos asociados	RF-16 Guardar el estado actual de las condiciones del invernadero
Descripción	El sistema deberá almacenar el estado de las condiciones actuales
Datos específicos	• Límite máximo sobrepasado • Condición normal • Límite mínimo no alcanzado
Intervalo temporal	Presente
Importancia	Importante
Urgencia	Alta
Estado	Validado
Estabilidad	Alta
Comentarios	La información referente al estado de las condiciones actuales del invernadero es atribuible a la aplicación de escritorio, aplicación Android y página web

Tabla 13 - Requisitos de Información - Información sobre el estado actual de las condiciones del invernadero

1.4 REQUISITOS FUNCIONALES

1.4.1 DEFINICIÓN DE ACTORES

ACT-01	**Usuario del sistema**
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Descripción	Este actor representa a la persona encargada de la supervisión del invernadero
Comentarios	ninguno

Tabla 14 - Actores - Usuario del sistema

ACT-02	**Dispositivo Arduino**
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Descripción	Este actor representa al dispositivo Arduino conectado en red
Comentarios	ninguno

Tabla 15 - Actores - Dispositivo Arduino

ACT-03	**Aplicación de escritorio**
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Descripción	Este actor representa a la aplicación de escritorio
Comentarios	ninguno

Tabla 16 - Actores - Aplicación de escritorio

ACT-04	**Aplicación Android**
Versión	1.0
Autores	Miguel Angel Hernanz Hernanz
Fuentes	n/a
Descripción	Este actor representa a la aplicación Android manejada por un usuario desde un dispositivo compatible
Comentarios	ninguno

Tabla 17 - Actores - Aplicación Android

1.4.2 DIAGRAMAS DE CASOS DE USO

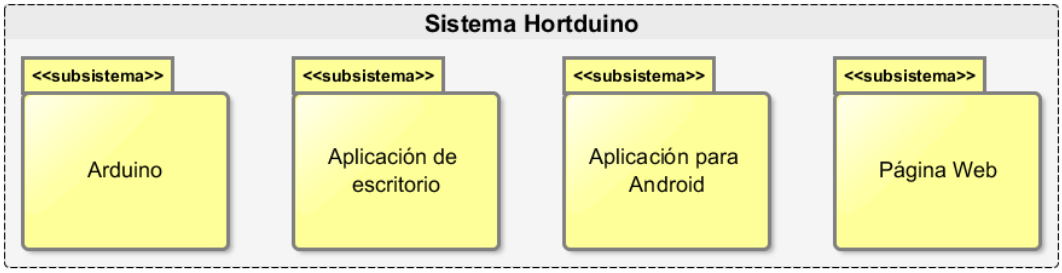


Ilustración 1 - Diagrama de subsistemas

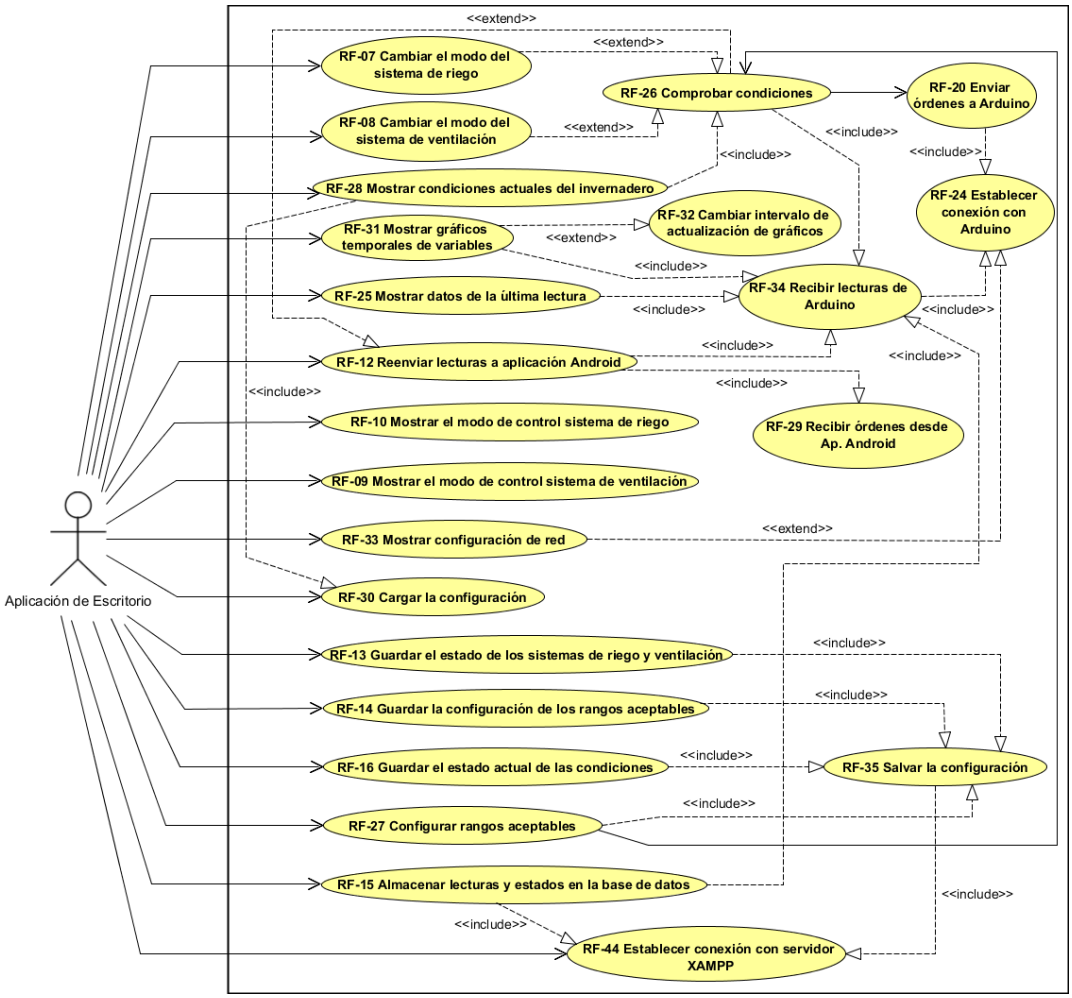


Ilustración 2 - Diagrama de casos de uso de la aplicación de escritorio

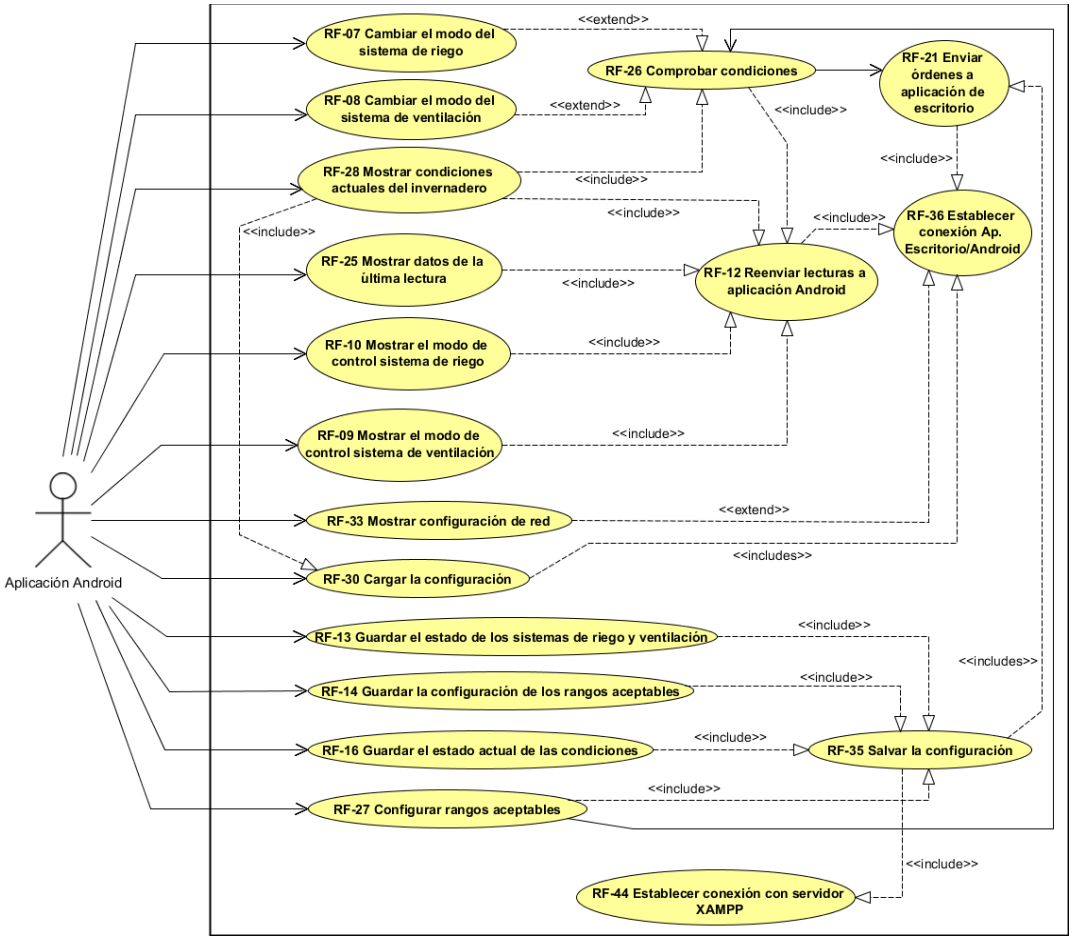


Ilustración 3 - Diagrama de casos de uso de la aplicación Android

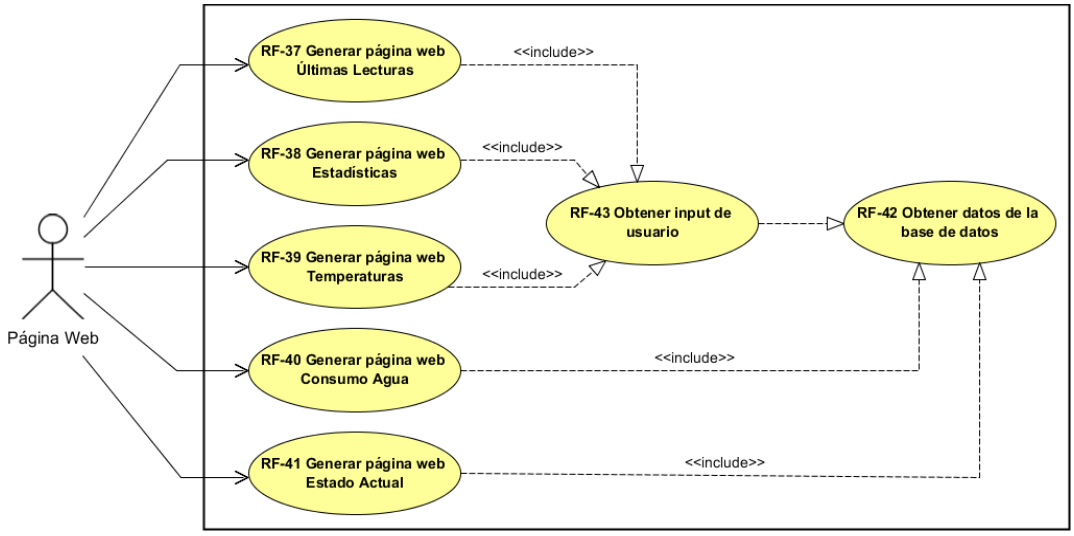


Ilustración 4 - Diagrama de casos de uso de la página Web

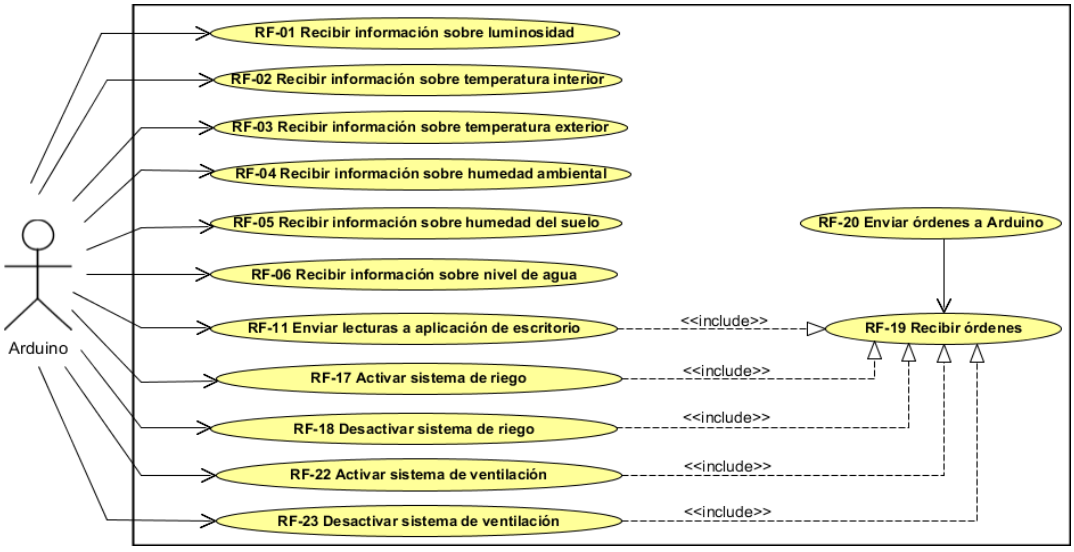


Ilustración 5 - Diagrama de casos de uso de Arduino

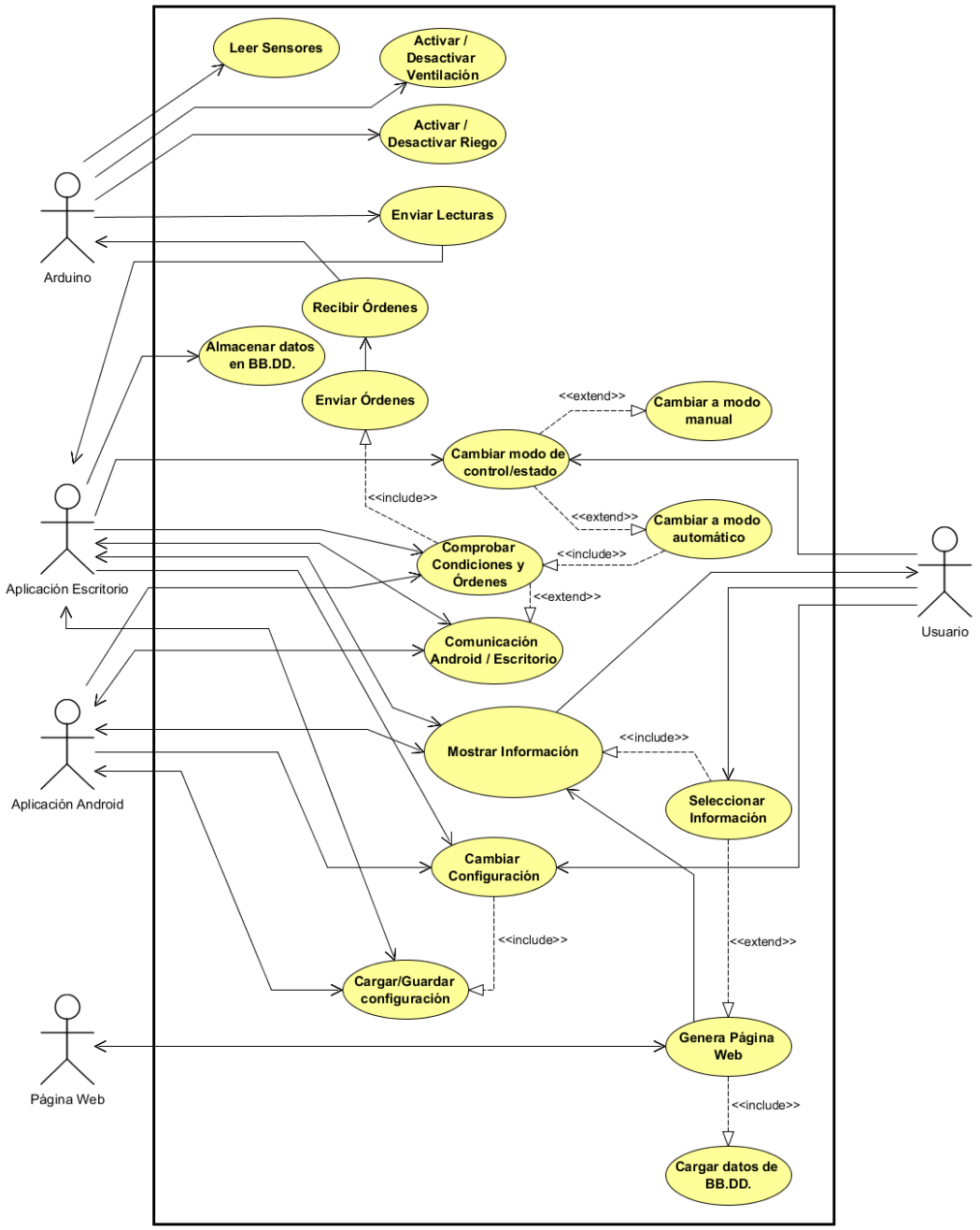


Ilustración 6 - Diagrama de casos de uso del sistema (simplificado)

1.4.3 CASOS DE USO DEL SISTEMA

RF-01	**Recibir información sobre luminosidad**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos	
Requisitos asociados	RI-01 Información procedente de los sensores remotos	
Descripción	Recibe en Arduino la información de luminosidad del sensor	
Precondición	Dispositivo Arduino en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Leer luminosidad en el pin de entrada asignado
	p2	Guardar el valor obtenido en la variable correspondiente
Postcondición	El valor de luminosidad está disponible para su envío	
Excepciones	**Paso**	**Acción**
	n/a	n/a
Rendimiento	Dentro del intervalo de tiempo asignado a la lectura de sensores	
Frecuencia esperada	Una lectura cada 1.5 segundos	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 18 - Caso de uso RF-01 Recibir información sobre luminosidad

RF-02	**Recibir información sobre temperatura interior**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos	
Requisitos asociados	RI-01 Información procedente de los sensores remotos	
Descripción	Recibe en Arduino la información de temperatura interior del sensor	
Precondición	Dispositivo Arduino en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Leer temperatura en el pin de entrada asignado
	p2	Guardar el valor obtenido en la variable correspondiente
Postcondición	El valor de temperatura interior está disponible para su envío	
Excepciones	**Paso**	**Acción**
	p1	Si el sensor enviá una lectura errónea (NaN), el valor asignado es 0
Rendimiento	Dentro del intervalo de tiempo asignado a la lectura de sensores	
Frecuencia esperada	Una lectura cada 1.5 segundos	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 19 - Caso de uso RF-02 Recibir información sobre temperatura interior

RF-03	**Recibir información sobre temperatura exterior**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos	
Requisitos asociados	RI-01 Información procedente de los sensores remotos	
Descripción	Recibe en Arduino la información de temperatura exterior del sensor	
Precondición	Dispositivo en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Leer temperatura exterior en el pin de entrada asignado y descartarla
	p2	Leer temperatura exterior en el pin de entrada y sumar este valor a la variable temporal
	p3	Esperar 100 milisegundos para estabilizar lecturas
	p4	Repetir los pasos 3 y 4 diez veces
	p5	Asignar el valor de la variable temporal dividido entre diez al resultado devuelto
Postcondición	El valor de temperatura exterior está disponible para su envío	
Excepciones	**Paso**	**Acción**
	n/a	n/a
Rendimiento	Dentro del intervalo de tiempo asignado a la lectura de sensores	
Frecuencia esperada	Una lectura cada 1.5 segundos	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 20 - Caso de uso RF-03 Recibir información sobre temperatura exterior

RF-04	**Recibir información sobre humedad ambiental**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos	
Requisitos asociados	RI-01 Información procedente de los sensores remotos	
Descripción	Recibe en Arduino la información de humedad ambiental del sensor	
Precondición	Dispositivo en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Leer humedad ambiental en el pin de entrada asignado
	p2	Guardar el valor obtenido en la variable correspondiente
Postcondición	El valor de humedad ambiental está disponible para su envío	
Excepciones	**Paso**	**Acción**
	p1	Si el sensor envía una lectura errónea (NaN), el valor asignado es 0
Rendimiento	Dentro del intervalo de tiempo asignado a la lectura de sensores	
Frecuencia esperada	Una lectura cada 1.5 segundos	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 21 - Caso de uso RF-04 Recibir información sobre humedad ambiental

RF-05	**Recibir información sobre humedad del suelo**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos	
Requisitos asociados	RI-01 Información procedente de los sensores remotos	
Descripción	Recibe en Arduino la información de humedad del suelo del sensor	
Precondición	Dispositivo en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Si se ha cumplido el intervalo de actualización, continuar
	p2	Activar el pin de salida a 5 voltios
	p3	Esperar 5 milisegundos para estabilizar
	p4	Leer valor recibido en el pin de entrada asignado
	p5	Desactivar el pin de salida con 0 voltios
Postcondición	El valor de temperatura exterior está disponible para su envío	
Excepciones	**Paso**	**Acción**
	p1	Si no se ha cumplido el intervalo de actualización, mantener la lectura anterior para su envío
Rendimiento	Dentro del intervalo de tiempo asignado a la lectura de sensores	
Frecuencia esperada	Una lectura cada 10 segundos	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	Lectura espaciada para evitar desgaste del sensor	

Tabla 22 - Caso de uso RF-05 Recibir información sobre humedad del suelo

RF-06	**Recibir información sobre nivel de agua**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos	
Requisitos asociados	RI-01 Información procedente de los sensores remotos	
Descripción	Recibe en Arduino la información de nivel de agua del sensor	
Precondición	Dispositivo en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Cambiar el modo del pin asignado a modo salida (OUTPUT)
	p2	Activar el pin de salida a 5 voltios durante 15 milisegundos
	p3	Desactivar el pin de salida con 0 voltios
	p4	Esperar 20 milisegundos
	p5	Cambiar el modo del pin asignado a modo entrada (INPUT)
	p6	Recibir el eco de retorno en el pin asignado
	p7	Calcular la distancia hasta la superficie del agua empleando el tiempo de retorno y la velocidad del sonido (29 cm/milisegundo)
	p8	Devolver en la variable correspondiente el valor porcentual del nivel de agua basándose en el valor obtenido en paso 6 y la profundidad del depósito
Postcondición	El porcentaje de nivel de agua restante está disponible para su envío	
Excepciones	**Paso**	**Acción**
	p8	Si el valor es negativo, se retorna el valor 0
Rendimiento	Dentro del intervalo de tiempo asignado a la lectura de sensores	
Frecuencia esperada	Una lectura cada 1.5 segundos	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 23 - Caso de uso RF-06 Recibir información sobre nivel de agua

RF-07	**Cambiar modo de control del sistema de riego**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-04 Cambiar el estado de los sistemas de riego/ventilación OBJ-06 Control automatizado de los sistemas de riego/ventilación	
Requisitos asociados	RF-26 Comprobar condiciones	
Descripción	Cambia el modo de control del sistema de riego alternando entre el modo manual (ON y OFF) y el modo automático (AUTO)	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión con servidor XAMPP Aplicación de escritorio/Android en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	El usuario hace click o pulsa el botón de control del sistema de riego
	p3	El estado del sistema de riego cambia
	p4	La nueva configuración se guarda en el archivo config.txt del servidor
	p5	Si el nuevo estado implica una activación o desactivación del sistema de riego basado en las condiciones impuestas por el usuario, las nuevas órdenes son enviadas a Arduino
Postcondición	El modo de control del sistema de riego ha cambiado El nuevo modo de control aparece en la interfaz del usuario	
Excepciones	**Paso**	**Acción**
	n/a	n/a
Rendimiento	De uno a cinco segundos en ser efectivo en Arduino	
Frecuencia esperada	Un cambio por segundo	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	El modo de control viene representado en la interfaz de usuario por el texto y el color del botón asociado a cada modo	

Tabla 24 - Caso de uso RF-07 Cambiar modo de control del sistema de riego

RF-08	**Cambiar modo de control del sistema de ventilación**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-04 Cambiar el estado de los sistemas de riego/ventilación OBJ-06 Control automatizado de los sistemas de riego/ventilación	
Requisitos asociados	RF-26 Comprobar condiciones	
Descripción	Cambia el modo de control del sistema de ventilación alternando entre el modo manual (ON y OFF) y el modo automático (AUTO)	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión con servidor XAMPP Aplicación de escritorio/Android en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	El usuario hace click o pulsa el botón de control del sistema de ventilación
	p3	El estado del sistema de ventilación cambia
	p4	La nueva configuración se guarda en el archivo config.txt del servidor
	p5	Si el nuevo estado implica una activación o desactivación del sistema de ventilación basado en las condiciones impuestas por el usuario, las nuevas órdenes son enviadas a Arduino
Postcondición	El modo de control del sistema de ventilación ha cambiado El nuevo modo de control aparece en la interfaz del usuario	
Excepciones	**Paso**	**Acción**
	n/a	n/a
Rendimiento	De uno a cinco segundos en ser efectivo en Arduino	
Frecuencia esperada	Un cambio por segundo	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	El modo de control viene representado en la interfaz de usuario por el texto y el color del botón asociado a cada modo	

Tabla 25 - Caso de uso RF-08 Cambiar modo de control del sistema de ventilación

RF-09	**Mostrar modo de control del sistema de ventilación**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-05 Mostrar el estado y modo de funcionamiento de los sistemas de riego y ventilación	
Requisitos asociados	RF-12 Reenviar lecturas a aplicación Android	
Descripción	Muestra el modo de control actual del sistema de ventilación, alternando entre el modo manual (ON y OFF) y el modo automático (AUTO)	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión con servidor XAMPP Aplicación de escritorio/Android en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	La aplicación carga la configuración
	p2	Se muestra el modo de control actual del sistema de ventilación
Postcondición	El modo de control del sistema de ventilación aparece en la interfaz	
Excepciones	**Paso**	**Acción**
	p1	Si no es posible cargar la configuración, se genera la configuración por defecto con valores igual a 0
Rendimiento	Instantáneo	
Frecuencia esperada	Un cambio por segundo	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	El modo de control viene representado en la interfaz de usuario por el texto y el color del botón asociado a cada modo	

Tabla 26 - Caso de uso RF-09 Mostrar modo de control del sistema de ventilación

RF-10	**Mostrar modo de control del sistema de riego**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-05 Mostrar el estado y modo de funcionamiento de los sistemas de riego y ventilación	
Requisitos asociados	RF-12 Reenviar lecturas a aplicación Android	
Descripción	Muestra el modo de control actual del sistema de riego, alternando entre el modo manual (ON y OFF) y el modo automático (AUTO)	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión con servidor XAMPP Aplicación de escritorio/Android en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	La aplicación carga la configuración
	p2	Se muestra el modo de control actual del sistema de riego
Postcondición	El modo de control del sistema de riego aparece en la interfaz	
Excepciones	**Paso**	**Acción**
	p1	Si no es posible cargar la configuración, se genera la configuración por defecto con valores igual a 0
Rendimiento	Instantáneo	
Frecuencia esperada	Un cambio por segundo	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	El modo de control viene representado en la interfaz de usuario por el texto y el color del botón asociado a cada modo	

Tabla 27 - Caso de uso RF-10 Mostrar modo de control del sistema de riego

RF-11	**Enviar lecturas a aplicación de escritorio**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos OBJ-02 Mostrar información de sensores remotos OBJ-03 Almacenar información recibida OBJ-04 Cambiar el estado de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-19 Recibir órdenes	
Descripción	Envía las lecturas efectuadas en los sensores a la aplicación de escritorio	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino y aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio/Android en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Si se ha sobrepasado el intervalo de actualización, Arduino lee los sensores. Si no se cumple la anterior condición, espera hasta su cumplimiento.
	p2	Arduino recopila la información de la lectura de sensores en una cadena de texto con valores numéricos separados por comas.
	p3	Si la aplicación de escritorio ha mandado órdenes, las lecturas son enviadas a la dirección IP desde la que se envió el paquete de órdenes.
	p4	Se actualiza el temporizador usado en el intervalo de actualización
Postcondición	Se recibe un paquete UDP en la aplicación de escritorio con los valores correspondientes a cada variable en una cadena de texto separados por comas	
Excepciones	**Paso**	**Acción**
	p3	Si no se han recibido órdenes, las lecturas de sensores son descartadas.
Rendimiento	En el intervalo fijado para su envío. Mínimo 1.5 segundos	
Frecuencia esperada	Un envío por intervalo fijado. 40 envíos por minuto máximo.	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 28 - Caso de uso RF-11 Enviar lecturas a aplicación de escritorio

RF-12	**Reenviar lecturas a aplicación Android**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos OBJ-02 Mostrar información de sensores remotos OBJ-04 Cambiar el estado de los sistemas de riego y ventilación OBJ-05 Mostrar el estado y modo de funcionamiento de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-34 Recibir lecturas de Arduino RF-29 Recibir órdenes desde aplicación Android RF-26 Comprobar condiciones	
Descripción	Reenvía las lecturas recibidas en la aplicación de escritorio a la ap. Android	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino y aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio/Android en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Se recibe un paquete solicitando el reenvío de las lecturas
	p2	El paquete con las lecturas enviado por Arduino es reenviado a la dirección IP que solicita las lecturas
Postcondición	Se recibe un paquete UDP en la aplicación Android con los valores correspondientes a cada variable en una cadena de texto separados por comas o bien el caracter 'c'	
Excepciones	**Paso**	**Acción**
	p2	Si la aplicación de escritorio produce un cambio en la configuración de modos de control y rangos aceptables cuando la aplicación Android está solicitando lecturas, se envía un paquete que contiene el caracter 'c' a la aplicación Android para que recargue la configuración.
Rendimiento	En el intervalo fijado para su recepción. Mínimo 1.5 segundos	
Frecuencia esperada	Un envío por intervalo fijado. 40 envíos por minuto máximo.	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 29 - Caso de uso RF-12 Reenviar lecturas a aplicación Android

RF-13	**Guardar el estado de los sistemas de riego y ventilación**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos OBJ-02 Mostrar información de sensores remotos OBJ-03 Almacenar información recibida OBJ-04 Cambiar el estado de los sistemas de riego y ventilación OBJ-05 Mostrar el estado y modo de funcionamiento de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RI-02 Información sobre el estado actual de los sistemas de riego y ventilación RI-03 Información sobre el método de control de los sistemas de riego y ventilación RF-35 Salvar configuración	
Descripción	Guarda el estado actual de los sistemas de riego y ventilación en un archivo con el formato (1=ON, 0=OFF, 2=AUTO)	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino y aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio/Android en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	La aplicación de escritorio construye una cadena de texto con las condiciones del usuario sobre los rangos aceptables de las variables monitorizadas y el modo de control de los sistemas de riego y ventilación.
	p2	La aplicación de escritorio escribe la cadena de texto en el archivo *config.txt*
Postcondición	El archivo *config.txt* contiene la configuración y el modo de control de los sistemas en una línea de texto con formato de separación de campos por punto y coma	
Excepciones	**Paso**	**Acción**
	p2	Si el archivo no existe, es creado. Si existe, se sobrescribe.
Rendimiento	Inferior a 5 segundos	
Frecuencia esperada	n/a	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	Este caso de uso es concurrente con RF-14	

Tabla 30 - Caso de uso RF-13 Guardar el estado de los sistemas de riego y ventilación

RF-14	**Guardar la configuración de los rangos aceptables de las variables monitorizadas**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-03 Almacenar información recibida OBJ-04 Cambiar el estado de los sistemas de riego y ventilación OBJ-05 Mostrar el estado y modo de funcionamiento de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RI-04 Información sobre la configuración de rangos aceptables de variables RF-35 Salvar configuración	
Descripción	Guarda la configuración actual de los rangos establecidos por el usuario para las variables monitorizadas	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino y aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio/Android en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	La aplicación de escritorio construye una cadena de texto con las condiciones del usuario sobre los rangos aceptables de las variables monitorizadas y el estado de los sistemas de riego y ventilación.
	p2	La aplicación de escritorio escribe la cadena de texto en el archivo *config.txt*
Postcondición	El archivo *config.txt* contiene la configuración y el modo de control de los sistemas en una línea de texto con formato de separación de campos por punto y coma	
Excepciones	**Paso**	**Acción**
	p2	Si el archivo no existe, es creado. Si existe, se sobrescribe.
Rendimiento	Inferior a 5 segundos	
Frecuencia esperada	n/a	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	Este caso de uso es concurrente con RF-13	

Tabla 31 - Caso de uso RF-14 Guardar la configuración de los rangos aceptables de las variables monitorizadas

RF-15	**Almacenar lecturas y estados en la base de datos**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos OBJ-02 Mostrar información de sensores remotos OBJ-03 Almacenar información recibida OBJ-05 Mostrar el estado y modo de funcionamiento de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RI-01 Información procedente de los sensores remotos RF-34 Recibir lecturas de Arduino RF-44 Establecer conexión con el servidor XAMPP	
Descripción	Guarda en la base de datos del servidor los datos de la lectura recibida y los estados actuales de los sistemas de riego y ventilación	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino y aplicación de escritorio Conexión con servidor XAMPP (servicio MySQL) Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Determinar si se ha cumplido el intervalo de guardado fijado
	p2	La aplicación de escritorio se autentifica con el gestor de la base de datos
	p3	Si hay conexión con la base de datos, se envía la consulta SQL para añadir un nuevo registro a la base de datos compuesto por la lectura recibida y el estado de los sistemas de riego y ventilación.
	p4	Se actualiza el temporizador usado en el intervalo de escritura
Postcondición	La base de datos del servidor contiene un nuevo registro con la lectura recibida y el estado de los sistemas de riego y ventilación en el momento de la recepción.	
Excepciones	**Paso**	**Acción**
	p2	Si no está disponible la conexión, se muestra un mensaje de error
Rendimiento	Inferior a 5 segundos	
Frecuencia esperada	n/a	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 32 - Caso de uso RF-15 Almacenar lecturas y estados en la base de datos

RF-16	**Guardar el estado actual de las condiciones del invernadero**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos OBJ-02 Mostrar información de sensores remotos OBJ-03 Almacenar información recibida OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RI-05 Información sobre el estado actual de las condiciones del invernadero RF-35 Salvar la configuración	
Descripción	Guarda en el archivo *condiciones.txt* es estado actual de las condiciones del invernadero con el siguiente formato: • -1 : No se alcanza el nivel mínimo establecido para la condición • 0: Situación normal • 1: Se ha superado el nivel máximo permitido para la condición	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino y aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Determinar si ha ocurrido un cambio en las condiciones actuales
	p2	La aplicación de escritorio construye una cadena de texto con el valor asociado al estado de cada condición, separado por comas, y el estado actual de los sistemas de riego y ventilación (activo o inactivo)
	p3	Se escribe la cadena en el archivo *condiciones.txt*
Postcondición	El archivo *condiciones.txt* contiene una cadena numérica con los valores representativos del estado de cada condición y de la actividad de los sistemas de riego y ventilación. La cadena debe contener un total de 6 comas separadoras.	
Excepciones	**Paso**	**Acción**
	p2	Si el archivo no existe, es creado. Si existe, se sobrescribe.
Rendimiento	Inferior a 5 segundos	
Frecuencia esperada	n/a	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 33 - Caso de uso RF-16 Guardar el estado actual de las condiciones del invernadero

RF-17	**Activar sistema de riego en Arduino**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-04 Cambiar estado de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-19 Recibir órdenes en Arduino RF-20 Enviar órdenes a Arduino	
Descripción	Activa el sistema de riego accionando el actuador mecánico asociado	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino y aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Se recibe un paquete UDP con la orden de activación del sistema de riego
	p2	Se activa el pin de salida asociado al actuador mecánico con 5 voltios
Postcondición	El sistema de riego está en funcionamiento	
Excepciones	**Paso**	**Acción**
	n/a	n/a
Rendimiento	Inferior a 5 segundos	
Frecuencia esperada	Dependiente de las condiciones ambientales	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 34 - Caso de uso RF-17 Activar sistema de riego en Arduino

RF-18	**Desactivar sistema de riego en Arduino**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-04 Cambiar estado de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación	
Requisitos asociados	RF-19 Recibir órdenes en Arduino RF-20 Enviar órdenes a Arduino	
Descripción	Desactiva el sistema de riego accionando el actuador mecánico asociado	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino y aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Se recibe un paquete UDP con la orden de desactivación del sistema de riego
	p2	Se desactiva el pin de salida asociado al actuador mecánico con 0 voltios
Postcondición	El sistema de riego no está en funcionamiento	
Excepciones	**Paso**	**Acción**
	n/a	n/a
Rendimiento	Inferior a 5 segundos	
Frecuencia esperada	Dependiente de las condiciones ambientales	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 35 - Caso de uso RF-18 Desactivar sistema de riego en Arduino

RF-19	**Recibir órdenes en Arduino**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-04 Cambiar estado de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-11 Enviar lecturas a aplicación de escritorio RF-17 Activar sistema de riego RF-18 Desactivar sistema de riego RF-22 Activar sistema de ventilación RF-23 Desactivar sistema de ventilación RF-20 Enviar órdenes a Arduino	
Descripción	Activa o desactiva los sistemas de riego y ventilación. Genera una petición de envío de lecturas a la IP asociada al paquete UDP.	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino y aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Si existe un paquete UDP disponible en el buffer de entrada, se copia su contenido en la variable que mantiene la última orden activa.
	p2	El primer byte del mensaje se descarta
	p3	El sistema de riego se activa (1) o desactiva (0) según el segundo byte del mensaje
	p4	El sistema de ventilación se activa (1) o desactiva (0) usando el tercer byte del mensaje
Postcondición	El sistema de riego está en funcionamiento	
Excepciones	**Paso**	**Acción**
	p1	Si no hay un paquete UDP disponible, se mantiene la última orden recibida
Rendimiento	Inferior a 5 segundos	
Frecuencia esperada	Cada 1.5 segundos	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 36 - Caso de uso RF-19 Recibir órdenes en Arduino

RF-20	**Enviar órdenes a Arduino**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-04 Cambiar estado de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-19 Recibir órdenes en Arduino RF-24 Establecer conexión con Arduino RF-26 Comprobar condiciones	
Descripción	Envía nuevas órdenes a la aplicación de escritorio Solicita el reenvío de las lecturas de sensores a la IP de origen del paquete UDP.	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino y aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Si el usuario cambia el modo de control a modo automático, se evalúan las condiciones fijadas por el usuario en las variables monitorizadas y se asigna el valor numérico correspondiente a la orden a enviar (1=ON, 0=OFF) para cada sistema.
	p2	Si el usuario cambia el modo de control a modo manual, se asigna el valor numérico correspondiente a la orden a enviar (1=ON, 0=OFF) para cada sistema.
	p3	La aplicación de escritorio construye el paquete UDP a enviar con tres bytes: el primero será el valor 1 y a continuación seguirán las órdenes para el sistema de riego (0/1) y el sistema de ventilación (0/1).
	p4	El paquete UDP con las órdenes es enviado a la IP de Arduino
Postcondición	Arduino recibe las órdenes a ejecutar. Arduino envía las lecturas de los sensores a la IP de la aplicación de escritorio	
Excepciones	**Paso**	**Acción**
	*	Si se han recibido órdenes en la aplicación de escritorio provenientes de la aplicación Android, se actualiza el estado y configuración del sistema de acuerdo a éstas, se guarda la configuración y se envían a Arduino
Rendimiento	Inferior a 5 segundos	
Frecuencia esperada	Cada 1.5 segundos	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 37 - Caso de uso RF-20 Enviar órdenes a Arduino

RF-21	**Enviar órdenes a aplicación de escritorio**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-04 Cambiar estado de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-29 Recibir órdenes desde Android RF-26 Comprobar condiciones RF-36 Establecer conexión ap. escritorio/Android RF-35 Salvar la configuración	
Descripción	Envía nuevas órdenes a la aplicación de escritorio desde la aplicación Android Solicita el reenvío de las lecturas de sensores a la aplicación Android.	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino y aplicación de escritorio Conexión con servidor XAMPP Aplicaciones de escritorio y Android en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Si el usuario cambia el modo de control a modo automático, se evalúan las condiciones fijadas por el usuario en las variables monitorizadas y se asigna el valor numérico correspondiente a la orden a enviar (1=ON, 0=OFF) para cada sistema.
	p2	Si el usuario cambia el modo de control a modo manual, se asigna el valor numérico correspondiente a la orden a enviar (1=ON, 0=OFF) para cada sistema.
	p3	La aplicación Android construye el paquete UDP a enviar con tres bytes: el primero indica si se ha modificado la configuración general en la aplicación Android (1) o simplemente se solicita el reenvío de las lecturas recibidas (0) y a continuación seguirán las órdenes para el sistema de riego (0/1) y el sistema de ventilación (0/1).
	p4	El paquete UDP con las órdenes es enviado a la aplicación de escritorio
Postcondición	Arduino recibe las órdenes a ejecutar. Arduino envía las lecturas de los sensores a la IP de la aplicación de escritorio	
Excepciones	**Paso**	**Acción**
	*	Si se han efectuado cambios en la configuración de la aplicación de escritorio, se actualiza el estado y configuración del sistema recargando la configuración en la aplicación Android.
Rendimiento	Inferior a 5 segundos	
Frecuencia esperada	Cada 1.5 segundos	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 38 - Caso de uso RF-21 Enviar órdenes a aplicación de escritorio

RF-22	**Activar sistema de ventilación en Arduino**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-04 Cambiar estado de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-19 Recibir órdenes en Arduino RF-20 Enviar órdenes a Arduino	
Descripción	Activa el sistema de ventilación accionando el actuador mecánico asociado	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión disponible con aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Se recibe un paquete UDP con la orden de activación del sistema de ventilación en Arduino
	p2	Se activa el pin de salida asociado al actuador mecánico con 5 voltios
Postcondición	El sistema de ventilación está en funcionamiento	
Excepciones	**Paso**	**Acción**
	n/a	n/a
Rendimiento	Inferior a 5 segundos	
Frecuencia esperada	Dependiente de las condiciones ambientales	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 39 - Caso de uso RF-22 Activar sistema de riego en Arduino

RF-23	**Desactivar sistema de ventilación en Arduino**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-04 Cambiar estado de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación	
Requisitos asociados	RF-19 Recibir órdenes en Arduino RF-20 Enviar órdenes a Arduino	
Descripción	Desactiva el sistema de ventilación accionando el actuador mecánico asociado	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión disponible con aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Se recibe un paquete UDP con la orden de desactivación del sistema de ventilación
	p2	Se desactiva el pin de salida asociado al actuador mecánico con 0 voltios
Postcondición	El sistema de ventilación no está en funcionamiento	
Excepciones	**Paso**	**Acción**
	n/a	n/a
Rendimiento	Inferior a 5 segundos	
Frecuencia esperada	Dependiente de las condiciones ambientales	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 40 - Caso de uso RF-23 Desactivar sistema de ventilación en Arduino

RF-24	**Establecer conexión con Arduino**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos OBJ-02 Mostrar información de sensores remotos OBJ-03 Almacenar información recibida OBJ-04 Cambiar estado de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-34 Recibir lecturas de Arduino RF-20 Enviar órdenes a Arduino RF-33 Mostrar configuración de red	
Descripción	Indica si hay comunicación disponible entre Arduino y la aplicación de escritorio	
Precondición	Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	La aplicación de escritorio envía paquetes UDP a la dirección IP de Arduino con las órdenes a ejecutar.
	p2	Se espera la respuesta del dispositivo Arduino en el socket de escucha UDP
	p3	La conexión se define activa si el número de paquetes esperados es menor o igual a dos
Postcondición	El sistema de riego no está en funcionamiento	
Excepciones	**Paso**	**Acción**
	p3	Si el número de paquetes esperados es mayor a dos, se muestra el menú de configuración de red
Rendimiento	Inferior a 3 segundos	
Frecuencia esperada	Cada intervalo de solicitud de lecturas / envío de órdenes	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 41 - Caso de uso RF-24 Establecer conexión con Arduino

RF-25	**Mostrar datos de lecturas recibidas**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos OBJ-02 Mostrar información de sensores remotos OBJ-05 Mostrar el estado y modo de funcionamiento de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-34 Recibir lecturas de Arduino	
Descripción	Muestra en la interfaz los datos de la última lectura recibida de Arduino	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión disponible con aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	La aplicación de escritorio recibe un paquete UDP conteniendo los valores de los sensores separados por comas
	p2	Si la aplicación Android está en funcionamiento, el paquete UDP es reenviado a la IP de la aplicación Android.
	p3	El contenido del paquete UDP es troceado y asignado a cada variable correspondiente usando el carácter coma como separador de valores.
	p4	Se muestra en pantalla el nombre de cada variable junto con el valor asignado
Postcondición	El usuario puede ver en la pantalla los datos de la última lectura recibida	
Excepciones	**Paso**	**Acción**
	p1	Si el paquete UDP no es de la longitud esperada, se sustituyen todos los datos por ceros y se muestra un error.
	p1	Si no se recibe ningún paquete, se muestra el menú de configuración de red o un mensaje de espera
Rendimiento	1.5 segundos	
Frecuencia esperada	Cada 1.5 segundos	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 42 - Caso de uso RF-25 Mostrar datos de lecturas recibidas

RF-26	**Comprobar condiciones**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-02 Mostrar información de sensores remotos OBJ-04 Cambiar estado de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-34 Recibir lecturas de Arduino RF-08 Cambiar el modo del sistema de ventilación RF-07 Cambiar el modo del sistema de riego RF-20 Enviar órdenes a Arduino RF-12 Reenviar lecturas a aplicación Android RF-27 Configurar rangos aceptables de las variables RF-28 Mostrar condiciones actuales del invernadero	
Descripción	Comprueba las condiciones actuales del invernadero para activar/desactivar los sistemas de riego y ventilación si se ha seleccionado el modo automático. Muestra de forma gráfica el estado de las condiciones en la interfaz del usuario	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión disponible con aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	La aplicación de escritorio compara el valor establecido por el usuario en los rangos mínimo y máximo de cada variable con el valor asociado a esa variable recibido en la última lectura
	p2	Si se cumplen las condiciones para activar o desactivar los sistemas de riego y ventilación y el modo está fijado en automático para ese sistema, se activa o desactiva según corresponda.
	p3	Asigna a las variables encargadas de mostrar el estado de las condiciones del invernadero el valor resultante de la comparación del paso 1 (ver descripción de los valores en el caso de uso RF-16)
	p4	Muestra en pantalla la información sobre cada condición usando el valor asociado a cada una en el paso anterior usando imágenes y texto
Postcondición	La aplicación genera una cadena de texto con el estado de las condiciones actuales. La aplicación detecta si se ha producido un cambio en las condiciones actuales.	
Excepciones	**Paso**	**Acción**
	n/a	n/a
Rendimiento	1.5 segundos	
Frecuencia esperada	Cada 1.5 segundos	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	Usado por el caso de uso RF-28 Mostrar condiciones actuales del invernadero	

Tabla 43 - Caso de uso RF-26 Comprobar condiciones

RF-27	**Configurar rangos aceptables**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-04 Cambiar estado de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-26 Comprobar condiciones RF-35 Salvar la configuración	
Descripción	Muestra la configuración de los rangos mínimo y máximo aceptables para cada variable. Permite la modificación del valor mediante botones para aumentar o disminuir el valor mostrado	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión disponible con aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	La aplicación muestra en pantalla los valores correspondientes a los rangos de cada variable
	p2	El usuario modificar ese valor mediante los botones situados a la derecha (en la aplicación escritorio) o arrastrar con el dedo el triángulo indicador sobre la barra (en la aplicación Android)
	p3	La aplicación guarda la nueva configuración (ver caso de uso RF-14)
Postcondición	Los nuevos rangos aceptables son aplicados y guardados en la configuración	
Excepciones	**Paso**	**Acción**
	n/a	n/a
Rendimiento	Instantáneo	
Frecuencia esperada	Indeterminado	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	Usado en los casos de uso RF-28 y RF-26	

Tabla 44 - Caso de uso RF-27 Configurar rangos aceptables

RF-28	**Mostrar condiciones actuales del invernadero**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos OBJ-02 Mostrar información de sensores remotos OBJ-04 Cambiar el estado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-26 Comprobar condiciones RF-30 Cargar configuración	
Descripción	Muestra en la interfaz la información disponible sobre las condiciones actuales, estados de los sistemas, últimos datos recibidos de los sensores y configuración de los rangos aceptables para cada variable.	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión disponible con aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	La aplicación carga la configuración
	p2	Se muestra en pantalla el estado de las condiciones actuales, nivel de agua restante en el depósito y los rangos fijados en la configuración
	p3	Se muestra en pantalla los botones y estados de los sistemas de r/v
	p4	Se muestra en pantalla la última lectura recibida desde Arduino
Postcondición	El usuario puede visualizar en pantalla la información indicada en la descripción	
Excepciones	**Paso**	**Acción**
	n/a	n/a
Rendimiento	Actualización cada 1.5 segundos	
Frecuencia esperada	En el intervalo de recepción de lecturas	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 45 - Caso de uso RF-28 Mostrar condiciones actuales del invernadero

RF-29	**Recibir órdenes desde Android**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos OBJ-04 Cambiar el estado de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-12 Reenviar lecturas a aplicación Android RF-26 Comprobar condiciones RF-35 Salvar la configuración RF-36 Establecer conexión aplicación de escritorio/Android RF-14 Guardar la configuración de los rangos aceptables de las variables	
Descripción	Envía el paquete con las órdenes a ejecutar en el dispositivo Arduino a la aplicación de escritorio.	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión disponible con aplicación de escritorio Conexión con servidor XAMPP Aplicaciones de escritorio y Android en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	La aplicación de escritorio detecta un paquete procedente de una IP distinta a la asignada al dispositivo Arduino
	p2	Se extraen del paquete recibido los tres bytes que lo componen
	p3	Si en el primer byte se encuentra un 1, se consideran que los dos bytes restantes son nuevas órdenes
	p4	Se asigna el valor del segundo byte al estado del sistema de riego
	p5	Se asigna el valor del tercer byte al estado del sistema de ventilación
	p6	La bandera de recarga de configuración se activa
Postcondición	El usuario puede visualizar en pantalla la información indicada en la descripción	
Excepciones	**Paso**	**Acción**
	p3	Si el valor del primer byte es 0, se reenvía la última lectura como en RF-12
Rendimiento	Actualización cada 1.5 segundos	
Frecuencia esperada	En el intervalo de recepción de lecturas	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 46 - Caso de uso RF-29 Recibir órdenes desde Android

RF-30	**Cargar la configuración**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-04 Cambiar el estado de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-28 Mostrar condiciones actuales del invernadero RF-36 Conexión aplicación de escritorio/Android	
Descripción	Carga la configuración de las condiciones y estados desde un archivo	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión disponible con aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	La aplicación lee la línea del archivo *config.txt* del servidor
	p2	Se trocea la línea anterior a un array usando como separador el carácter “;”
	p3	Se asigna a cada variable monitorizada el valor del rango máximo y mínimo que resulta de la descomposición de la cadena del paso 2
	p4	Se asigna el valor correspondiente al estado del sistema de riego contenido en la cadena de texto del paso 2
	p5	Se asigna el valor correspondiente al estado del sistema de ventilación contenido en la cadena de texto del paso 2
Postcondición	La configuración ha sido cargada en la aplicación	
Excepciones	**Paso**	**Acción**
	p1	Si no existe el archivo o la configuración no es válida, se asigna el valor 0 a todas las condiciones y se crea el archivo (aplicación de escritorio) o se finaliza la aplicación (aplicación Android)
Rendimiento	Menor a 3 segundos	
Frecuencia esperada	No definido	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	Este caso de uso es común en la aplicación de escritorio y Android	

Tabla 47 - Caso de uso RF-30 Cargar la configuración

RF-31	**Mostrar gráficos temporales de las variables**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-32 Cambiar intervalo de actualización de gráficos RF-34 Recibir lecturas de Arduino	
Descripción	Muestra los gráficos de evolución temporal reciente de las variables monitorizadas en la aplicación de escritorio	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión disponible con aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p0	Si se ha cumplido el intervalo de actualización de gráficos, se procede al paso 1. En caso contrario se espera hasta que se alcance.
	p1	La aplicación recibe la lectura de los sensores y almacena cada valor recibido en el array correspondiente a cada variable en la posición indicada por el índice actual. Se incrementa el índice con cada recepción de datos.
	p2	Se trazan los ejes en las posiciones predefinidas para los dos gráficos
	p3	Para cada variable monitorizada, se recorre el array correspondiente desde la posición inicial hasta el índice actual trazando curvas (splines) en las coordenadas calculadas con la función de ajuste al gráfico.
	p4	Si el índice actual no ha alcanzado el tamaño máximo fijado para el gráfico, se dibuja un punto del color del gráfico en el extremo del gráfico.
	p5	Si el índice ha alcanzado el tamaño máximo del gráfico, se reinicializa a cero para apuntar al inicio del array.
Postcondición	Se dibujan los gráficos y ejes correspondientes a cada variable en la interfaz	
Excepciones	**Paso**	**Acción**
	p1	Si el indice supera el tamaño del array, el índice se reinicializa a cero
	p4	Si el índice ha superado el tamaño máximo fijado para el gráfico, se dibuja un punto de color blanco en la posición actual de dibujado para indicar la posición actual de redibujado del gráfico
Rendimiento	La frecuencia de dibujado del gráfico es modificable	
Frecuencia esperada	No definido	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 48 - Caso de uso RF-31 Mostrar gráficos temporales de variables

RF-32	**Cambiar intervalo de actualización de gráficos**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-31 Mostrar gráficos temporales de las variables	
Descripción	Cambia el intervalo de dibujado/actualización de los gráficos temporales	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión disponible con aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Se muestra en pantalla los selectores de actualización para cada unidad de tiempo (horas, minutos y segundos)
	p2	El usuario modifica el intervalo usando los botones de incremento y decremento en cada unidad de tiempo mostrada.
	p3	Se convierte cada intervalo modificado a milisegundos y se combinan en la variable encargada de controlar si se ha cumplido el intervalo de actualización de gráficos.
Postcondición	La configuración de actualización de gráficos temporales ha sido cambiada	
Excepciones	**Paso**	**Acción**
	N/a	N/a
Rendimiento	No definido	
Frecuencia esperada	No definido	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 49 - Caso de uso RF-32 Cambiar intervalo de actualización de gráficos

RF-33	**Mostrar configuración de red**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos OBJ-02 Mostrar información de sensores remotos OBJ-04 Cambiar el estado de los sistemas de riego y ventilación OBJ-05 Mostrar el estado y modo de funcionamiento de los sistemas de riego y ventilación OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-36 Establecer conexión aplicación de escritorio/Android RF-24 Establecer conexión con Arduino	
Descripción	Muestra la ventana de configuración de red al usuario	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión disponible con aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	Se muestra en pantalla la dirección IP usada para conectar a Arduino
	p2	El usuario introduce la nueva dirección IP y guarda los cambios pulsando ENTER. La nueva dirección IP introducida es usada para conectar.
	p3	Se muestra en pantalla el puerto usado para conectar a Arduino
	p4	El usuario introduce el nuevo puerto y guarda los cambios pulsando ENTER. El nuevo puerto se usa en la conexión.
Postcondición	Los datos usados para conectar a Arduino han sido cambiados (ap. Escritorio) Se muestra la pantalla de espera con la información de conexión (Android)	
Excepciones	**Paso**	**Acción**
	p2	En la aplicación Android, no se puede modificar la IP
	p4	En la aplicación Android, no se puede modificar el puerto
Rendimiento	Si el contador de paquetes esperados supera el límite de 2 (3 segundos)	
Frecuencia esperada	No definido	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 50 - Caso de uso RF-33 Mostrar configuración de red

RF-34	**Recibir lecturas de Arduino**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos OBJ-02 Mostrar información de sensores remotos OBJ-05 Mostrar el estado y modo de funcionamiento de los sistemas de r/v OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar sobre las condiciones actuales del invernadero	
Requisitos asociados	RF-26 Comprobar condiciones RF-25 Mostrar datos de lecturas recibidas RF-24 Establecer conexión con Arduino RF-31 Mostrar gráficos temporales de las variables RF-12 Reenviar lecturas a aplicación Android RF-15 Almacenar lecturas y estados en la base de datos	
Descripción	Recibe las lecturas de los sensores enviadas por Arduino en la aplicación de escritorio	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión disponible con aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	La aplicación de escritorio mantiene un socket de escucha UDP en el puerto especificado.
	p2	Si recibe un paquete UDP procedente de la dirección IP de Arduino, se continua en el siguiente paso.
	p3	Si el tamaño del paquete UDP es correcto, se convierte la información contenida a una cadena de texto (se utiliza en otros procedimientos).
	p4	La cadena de texto es troceada usando el carácter “,” como separador de valores. Los valores son incorporados a los arrays de las variables en la posición correspondiente para dibujar los gráficos temporales (ver RF-31)
	p5	Se marca la conexión como activa y se reinicializa el contador de paquetes esperados a 0.
Postcondición	Los datos de los sensores enviados por Arduino son recibidos correctamente en la aplicación de escritorio	
Excepciones	**Paso**	**Acción**
	p2	Si el paquete procede de otra dirección IP, se reenvía la última lectura
	p3	Si el tamaño no es correcto, se usa el valor 0 en todos los campos
Rendimiento	1.5 segundos	
Frecuencia esperada	Una lectura recibida cada 1.5 segundos como máximo	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 51 - Caso de uso RF-34 Recibir lecturas de Arduino

RF-35	**Salvar la configuración**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Objetivos asociados	OBJ-04 Cambiar el estado de los sistemas de riego y ventilación OBJ-05 Mostrar el estado y modo de funcionamiento de los sistemas de r/v OBJ-06 Control automatizado de los sistemas de riego y ventilación	
Requisitos asociados	RI-03 Información sobre el método de control de los sistemas de riego y ventilación RI-04 Información sobre la configuración de rangos aceptables de variables RI-05 Información sobre el estado actual de las condiciones del invernadero RF-13 Guardar el estado de los sistemas de riego y ventilación RF-14 Guardar la configuración de los rangos aceptables RF-16 Guardar el estado actual de las condiciones RF-27 Configurar rangos aceptables RF-44 Establecer conexión con el servidor XAMPP RF-21 Enviar órdenes a aplicación escritorio	
Descripción	Guarda la configuración y estado del sistema en el servidor XAMPP	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión disponible con aplicación de escritorio Conexión con servidor XAMPP Aplicación de escritorio en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	La aplicación de escritorio recopila la información relativa a los rangos aceptables de las variables monitorizadas en una cadena de texto. Cada valor se separa usando el caracter “;” y ocupa una determinada posición dentro de esta cadena.
	p2	Se añaden los códigos del modo de control de los sistemas de riego y ventilación a la cadena del paso 1 de acuerdo con el formato numérico recogido en la descripción del caso de uso RF-13
	p3	La cadena de texto construida en los pasos 1 y 2 se guarda en el archivo *config.txt* en el servidor XAMPP
	p4	La aplicación de escritorio reúne los códigos de estado de las condiciones del invernadero separados por comas en una cadena de texto.
	p5	Se añade el estado actual de los sistemas de riego y ventilación con el código de estado asociado a cada uno (0=inactivo, 1=activo)
	p6	La cadena de texto construida en los pasos 4 y 5 se guarda en el archivo *condiciones.txt* en el servidor XAMPP
Postcondición	La configuración y el estado actual del sistema se ha guardado en los archivos correspondientes del servidor XAMPP	
Excepciones	**Paso**	**Acción**
	3 y 4	Si el archivo no existe, se crea. Si existe, el contenido es sobrescrito
Rendimiento	5 segundos máximo	
Frecuencia esperada	Con cada cambio de condiciones o configuración	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	

Tabla 52 - Caso de uso RF-35 Salvar la configuración

RF-36	**Establecer conexión ap. escritorio/Android**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-01 Recibir información de sensores remotos OBJ-02 Mostrar información de sensores remotos OBJ-04 Cambiar el estado de los sistemas de riego y ventilación OBJ-05 Mostrar el estado y modo de funcionamiento de los sistemas de r/v OBJ-06 Control automatizado de los sistemas de riego y ventilación OBJ-08 Informar de las condiciones actuales del invernadero	
Requisitos asociados	RF-12 Reenviar lecturas a aplicación Android RF-21 Enviar órdenes a aplicación de escritorio RF-30 Cargar la configuración RF-33 Mostrar configuración de red	
Descripción	Indica si la conexión entre la aplicación de escritorio y la aplicación de Android está activa.	
Precondición	Dispositivo Arduino en funcionamiento Conexión disponible con Arduino Conexión disponible con aplicación de escritorio Conexión con servidor XAMPP Aplicaciones de escritorio y Android en funcionamiento	
Secuencia normal	**Paso**	**Acción**
	p1	La aplicación Android lee la dirección IP a la que conectarse desde el archivo *IP.txt* del servidor XAMPP.
	p2	Se envía un paquete UDP a la dirección obtenida en el paso 1
	p3	Se recibe un paquete UDP procedente de la dirección IP con la última lectura recibida de los sensores.
	p4	Se reinicializa a 0 el contador de paquetes esperados y se marca como activa la conexión.
Postcondición	La configuración y el estado actual del sistema se ha guardado en los archivos correspondientes del servidor XAMPP	
Excepciones	**Paso**	**Acción**
	p1	Si el archivo no existe, se aborta la ejecución de la aplicación Android
	p3	Si no se reciben paquetes UDP en el intervalo esperado, se incrementa el contador de paquetes esperados.
	p4	Si el contador de paquetes esperados supera el valor 2, se marca la conexión como inactiva y se muestra en pantalla la configuración de red.
Rendimiento	3 segundos	
Frecuencia esperada	Con cada recepción de lecturas	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 53 - Caso de uso RF-36 Establecer conexión ap. escritorio/Android

RF-37	**Generar página web Ultimas Lecturas**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-07 Consulta de información vía web OBJ-08 Informar de las condiciones actuales del invernadero	
Requisitos asociados	RF-43 Obtener input de usuario	
Descripción	Genera la página web con la información de las últimas lecturas en forma de gráficos ajustables por el usuario	
Precondición	Conexión con servidor XAMPP (servicios MySQL y servidor web Apache)	
Secuencia normal	**Paso**	**Acción**
	p1	El usuario accede con su navegador a la sección Ultimas Lecturas de la web
	p2	Se genera la página web en el servidor usando el valor por defecto (1 hora) y se envía al navegador del usuario.
	p3	El usuario puede variar el intervalo temporal a mostrar en los gráficos seleccionando uno de los valores en la lista desplegable. Esto genera una petición al servidor web con el nuevo intervalo elegido.
	p4	El servidor consulta a la base de datos para extraer los datos requeridos por el usuario y los envía al navegador.
Postcondición	La página web solicitada se muestra en el navegador del usuario.	
Excepciones	**Paso**	**Acción**
	p2	Si se produce un error al consultar la base de datos, se muestra en pantalla
Rendimiento	Indeterminado	
Frecuencia esperada	Indeterminado	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	El usuario puede desplazar el cursor del ratón por encima de los gráficos para obtener valores mas precisos.	

Tabla 54 - Caso de uso RF-37 Generar página web Ultimas Lecturas

RF-38	**Generar página web Estadísticas**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-07 Consulta de información vía web OBJ-08 Informar de las condiciones actuales del invernadero	
Requisitos asociados	RF-43 Obtener input de usuario	
Descripción	Genera la página web con la información gráfica sobre la evolución de las variables monitorizadas y el estado de los sistemas a lo largo de un periodo de tiempo modificable por el usuario.	
Precondición	Conexión con servidor XAMPP (servicios MySQL y servidor web Apache)	
Secuencia normal	**Paso**	**Acción**
	p1	El usuario accede con su navegador a la sección Estadísticas de la web
	p2	Se genera la página web en el servidor usando el valor por defecto (medio día) y se envía al usuario.
	p3	El usuario puede variar el intervalo temporal a mostrar en el gráfico de estadísticas, así como seleccionar las variables a mostrar. Esto genera una petición al servidor web con las opciones escogidas por el usuario.
	p4	El servidor consulta a la base de datos para extraer los datos requeridos por el usuario y los envía al navegador.
Postcondición	La página web solicitada se muestra en el navegador del usuario.	
Excepciones	**Paso**	**Acción**
	p2	Si se produce un error al consultar la base de datos, se muestra en pantalla
Rendimiento	Indeterminado	
Frecuencia esperada	Indeterminado	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 55 - Caso de uso RF-38 Generar página web Estadísticas

RF-39	**Generar página web Temperaturas**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-07 Consulta de información vía web OBJ-08 Informar de las condiciones actuales del invernadero	
Requisitos asociados	RF-43 Obtener input de usuario	
Descripción	Genera la página web con la información disponible sobre la evolución de las temperaturas del invernadero en un intervalo modificable por el usuario.	
Precondición	Conexión con servidor XAMPP (servicios MySQL y servidor web Apache)	
Secuencia normal	**Paso**	**Acción**
	p1	El usuario accede con su navegador a la sección Temperaturas de la web
	p2	Se genera la página web en el servidor usando el valor por defecto (un día) y se envía al usuario
	p3	El usuario puede variar el intervalo temporal a mostrar introduciendo un valor en el campo *Número de* días. Esto genera una petición al servidor web con el intervalo de días escogido por el usuario.
	p4	El servidor consulta a la base de datos para extraer los datos requeridos por el usuario y los envía al navegador.
Postcondición	La página web solicitada se muestra en el navegador del usuario.	
Excepciones	**Paso**	**Acción**
	p2	Si se produce un error al consultar la base de datos, se muestra en pantalla
Rendimiento	Indeterminado	
Frecuencia esperada	Indeterminado	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 56 - Caso de uso RF-39 Generar página web Temperaturas

RF-40	**Generar página web Consumo de Agua**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-07 Consulta de información vía web OBJ-08 Informar de las condiciones actuales del invernadero	
Requisitos asociados	RF-42 Obtener datos de la base de datos	
Descripción	Genera la página web con la información disponible sobre el nivel actual de agua en el depósito de riego, el consumo total y medio y la previsión de duración del agua para riego en base al consumo y el nivel restante.	
Precondición	Conexión con servidor XAMPP (servicios MySQL y servidor web Apache)	
Secuencia normal	**Paso**	**Acción**
	p1	El usuario accede con su navegador a la sección Consumo de Agua de la web
	p2	Se genera la página web en el servidor usando los datos obtenidos de la base de datos.
	p3	Se envía la página web al navegador del usuario
Postcondición	La página web solicitada se muestra en el navegador del usuario.	
Excepciones	**Paso**	**Acción**
	p2	Si se produce un error al consultar la base de datos, se muestra en pantalla
Rendimiento	Indeterminado	
Frecuencia esperada	Indeterminado	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 57 - Caso de uso RF-40 Generar página web Consumo de Agua

RF-41	**Generar página web Estado Actual**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-07 Consulta de información vía web OBJ-08 Informar de las condiciones actuales del invernadero	
Requisitos asociados	RF-43 Obtener estados y condiciones	
Descripción	Genera la página web con la información disponible sobre el nivel actual de agua en el depósito de riego, el consumo total y medio y la previsión de duración del agua para riego en base al consumo y el nivel restante.	
Precondición	Conexión con servidor XAMPP (servicios MySQL y servidor web Apache)	
Secuencia normal	**Paso**	**Acción**
	p1	El usuario accede con su navegador a la sección Estado Actual de la web
	p2	El servidor lee las condiciones actuales del archivo *condiciones.txt* y la configuración del archivo *config.txt*
	p3	El servidor asocia a cada condición y estado la imagen correspondiente en función del código numérico asignado.
	p4	El servidor obtiene los estados y las condiciones de la última lectura registrada en la base de datos.
	p5	La página web generada con toda la información de los pasos anteriores se envía al navegador del usuario.
Postcondición	La página web solicitada se muestra en el navegador del usuario.	
Excepciones	**Paso**	**Acción**
	p2	Si se produce un error al consultar la base de datos, se muestra en pantalla
Rendimiento	Indeterminado	
Frecuencia esperada	Indeterminado	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	Esta página se autorefresca cada 30 segundos para obtener la última información recibida en el sistema.	

Tabla 58 - Caso de uso RF-41 Generar página web Estado Actual

RF-42	**Obtener datos de la base de datos**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-07 Consulta de información vía web OBJ-08 Informar de las condiciones actuales del invernadero	
Requisitos asociados	RF-40 Generar página web Consumo de Agua RF-41 Generar página web Estado Actual RF-43 Obtener input de usuario	
Descripción	Realiza una consulta en formato SQL a la base de datos y devuelve la información requerida por el usuario a través de la página web.	
Precondición	Conexión con servidor XAMPP (servicios MySQL y servidor web Apache)	
Secuencia normal	**Paso**	**Acción**
	p1	La página web que visita el usuario solicita información de la base de datos para mostrar en el navegador.
	p2	Se genera una consulta SQL que es enviada al servicio MySQL del servidor XAMPP
	p3	Se procesa la consulta por el motor MySQL y el resultado es devuelto a la página web que lo solicitaba.
Postcondición	Los datos requeridos son enviados al destino (página web o sketch)	
Excepciones	**Paso**	**Acción**
	p3	Si se produce un error al consultar la base de datos, se muestra en pantalla
Rendimiento	Indeterminado	
Frecuencia esperada	Indeterminado	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	n/a	

Tabla 59 - Caso de uso RF-42 Obtener datos de la base de datos

RF-43	**Obtener input de usuario**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-07 Consulta de información vía web OBJ-08 Informar de las condiciones actuales del invernadero	
Requisitos asociados	RF-37 Generar página web Ultimas Lecturas RF-38 Generar página web Estadísticas RF-39 Generar página web Temperaturas	
Descripción	Obtiene la entrada del usuario en la página web	
Precondición	Conexión con servidor XAMPP (servicios MySQL y servidor web Apache)	
Secuencia normal	**Paso**	**Acción**
	p1	El usuario accede con su navegador a la página web
	p2	El usuario introduce un valor numérico en el campo dispuesto, elige una opción de la lista desplegable o bien marca una o varias casillas de selección, según corresponda.
	p3	El input del usuario se recoge y utiliza en la construcción de la consulta SQL que será enviada desde la página web al servidor XAMPP
Postcondición	La opción elegida por el usuario es usada para mostrar la página web respectiva.	
Excepciones	**Paso**	**Acción**
	N/a	N/a
Rendimiento	Indeterminado	
Frecuencia esperada	Indeterminado	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	Se considera *input* a toda entrada que el usuario pueda efectuar en la página web	

Tabla 60 - Caso de uso RF-43 Obtener input de usuario

RF-44	**Establecer conexión con servidor XAMPP**	
Versión	1.0	
Autores	Miguel Angel Hernanz Hernanz	
Fuentes	n/a	
Objetivos asociados	OBJ-03 Almacenar la información recibida OBJ-07 Consulta de información vía web	
Requisitos asociados	RF-15 Almacenar lecturas y estados en la base de datos RF-35 Guardar configuración	
Descripción	Obtiene la entrada del usuario en la página web	
Precondición	-	
Secuencia normal	**Paso**	**Acción**
	p1	La aplicación de escritorio solicita acceso a un archivo de configuración del servidor.
	p2	El servidor XAMPP responde afirmativamente y concede acceso.
Postcondición	La aplicación de escritorio y página web tienen acceso al servidor XAMPP y pueden desarrollar sus funciones.	
Excepciones	**Paso**	**Acción**
	p2	Si el servidor XAMPP no responde, ninguna aplicación se ejecuta
Rendimiento	Indeterminado	
Frecuencia esperada	Indeterminado	
Importancia	Alta	
Urgencia	Alta	
Estado	Validado	
Estabilidad	Alta	
Comentarios	El servidor XAMPP es, junto a la aplicación de escritorio, uno de los componentes críticos que son imprescindibles para el funcionamiento de Hortduino	

Tabla 61 - Caso de uso RF-44 Establecer conexión con servidor XAMPP

1.5 REQUISITOS NO FUNCIONALES

RNF-01	Interoperabilidad
Objetivos asociados	-
Requisitos asociados	-
Descripción	El sistema debe contar con una herramienta que proporcione movilidad e independencia al usuario y que contenga las funciones esenciales de control y supervisión del invernadero, complementándose con la aplicación principal. La configuración de los sistemas del invernadero y los parámetros de control ha de ser persistente entre sesiones y aplicaciones.
Comentarios	-

Tabla 62 - Requisito no funcional RNF-01 Interoperabilidad

RNF-02	Accesibilidad de la información
Objetivos asociados	-
Requisitos asociados	-
Descripción	El sistema debe suministrar la información requerida por los usuarios en medios que puedan ser de fácil acceso simultáneo, como una página web. La información debe permanecer disponible por tiempo indeterminado para futuras consultas.
Comentarios	-

Tabla 63 - Requisito no funcional RNF-02 Accesibilidad

RNF-03	Diseño de la interfaz de usuario
Objetivos asociados	-
Requisitos asociados	-
Descripción	El sistema debe proporcionar herramientas con interfaces sencillas de manejar pero que contengan las funcionalidades requeridas de control y supervisión.
Comentarios	-

Tabla 64 - Requisito no funcional RNF-03 Diseño de la interfaz de usuario

RNF-04	Respuesta rápida
Objetivos asociados	-
Requisitos asociados	-
Descripción	El sistema debe ser capaz de responder con rapidez a los cambios detectados en las condiciones del invernadero y aplicar los cambios que el usuario considera oportunos.
Comentarios	-

Tabla 65 - Requisito no funcional RNF-04 Respuesta rápida

1.6 MATRIZ DE RASTREABILIDAD OBJETIVOS/REQUISITOS

	OBJ-01	OBJ-02	OBJ-03	OBJ-04	OBJ-05	OBJ-06	OBJ-07	OBJ-08
RI-01	●		●					
RI-02					●			
RI-03						●		
RI-04						●		●
RI-05								●
RF-01	●							
RF-02	●							
RF-03	●							
RF-04	●							
RF-05	●							
RF-06	●							
RF-07				●		●		
RF-09					●			
RF-10					●			
RF-11	●	●	●	●		●		●
RF-12	●	●		●	●	●		●
RF-13	●	●	●	●	●	●		●
RF-14			●	●	●	●		●
RF-15	●	●	●		●			●
RF-16	●	●	●					●
RF-17				●		●		●
RF-18				●		●		
RF-19				●		●		●
RF-20				●		●		●
RF-21				●		●		●
RF-22				●		●		●
RF-23				●		●		
RF-24	●	●	●	●		●		●
RF-25	●	●			●			●
RF-26		●		●		●		●
RF-27				●		●		●
RF-28	●	●		●				●
RF-29	●			●		●		●

Tabla 66 - Matriz de rastreabilidad objetivos/requisitos (1/2)

	OBJ-01	OBJ-02	OBJ-03	OBJ-04	OBJ-05	OBJ-06	OBJ-07	OBJ-08
RF-30				●		●		●
RF-31								●
RF-32								●
RF-33	●	●		●	●	●		●
RF-34	●	●			●	●		●
RF-35				●	●	●		
RF-36	●	●		●	●	●		●
RF-37							●	●
RF-38							●	●
RF-39							●	●
RF-40							●	●
RF-41							●	●
RF-42							●	●
RF-43							●	●
RF-44			●				●	
RNF-01								
RNF-02								
RNF-03								
RNF-04								

Tabla 67 - Matriz de rastreabilidad objetivos/requisitos (2/2)

2 ESPECIFICACIÓN FUNCIONAL DEL SISTEMA

2.1 MODELO DE PROCESOS DEL SISTEMA

En este apartado se abordan las funciones del sistema desde el punto de vista del desarrollador, donde se define qué debe hacer el sistema internamente para cumplir los objetivos y funciones recogidos en los requisitos.

En primer lugar se presentan los Diagramas de Flujo de Datos (DFD) en los que se muestra la circulación de la información dentro del sistema. En el Diagrama de Flujo de Datos de nivel 0, también llamado Diagrama de Contexto, se indican las interacciones que realiza el sistema con las entidades externas del entorno, ofreciendo una visión general del sistema a nivel relacional. Para detallar mas en profundidad los diversos subsistemas y el intercambio de información entre ellos, se emplean Diagramas de Flujo de Datos a mas bajo nivel, útiles para apreciar qué relaciones son establecidas y la información con la que trabaja cada elemento de forma mas específica.

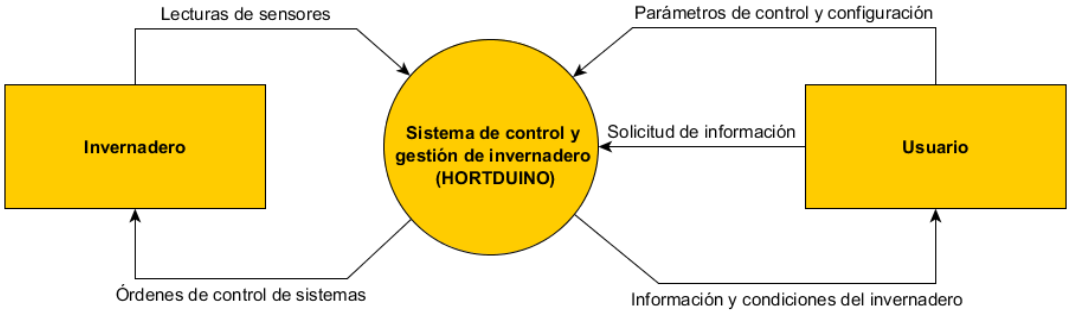


Ilustración 7 - Diagrama de Contexto del sistema - Nivel 0

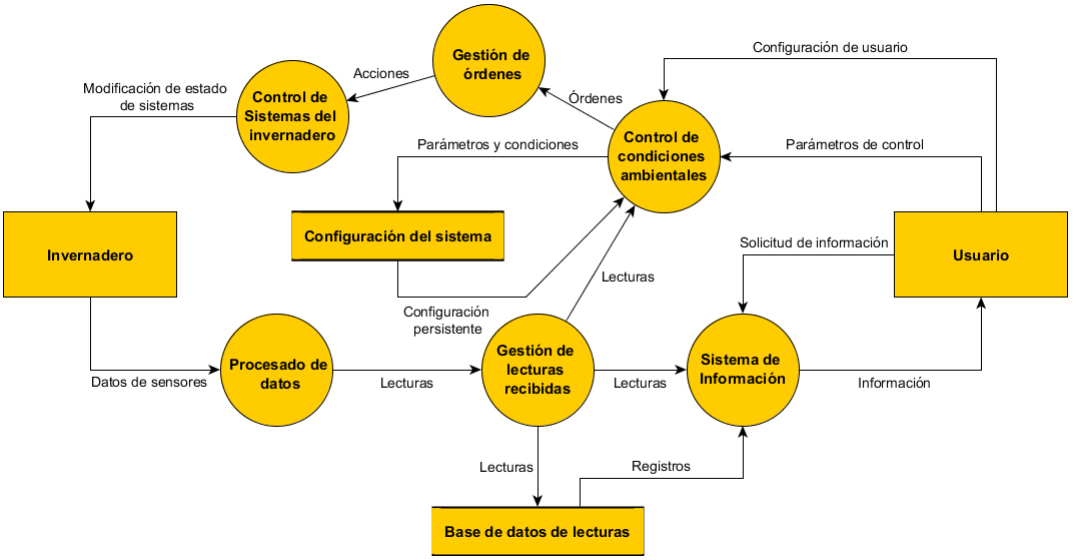


Ilustración 8 - Diagrama de Flujo de Datos - Nivel 1

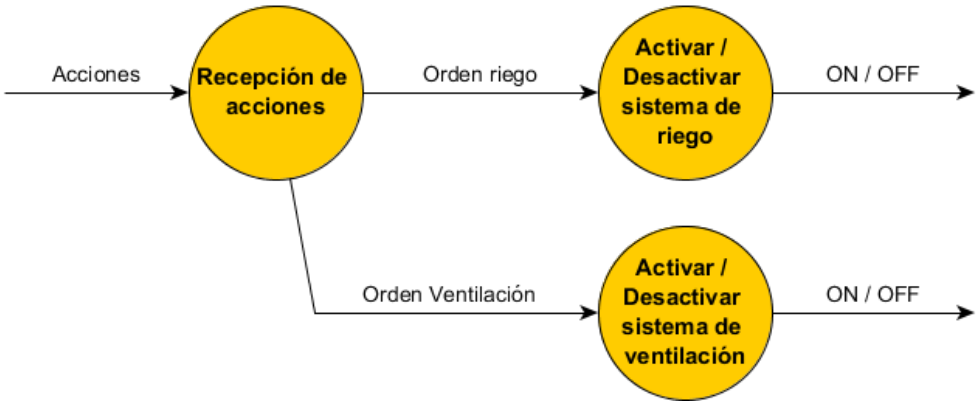


Ilustración 9 - Diagrama de Flujo de Datos - Nivel 2 - Control de los sistemas del invernadero

Ilustración 10 - Diagrama del Flujo de Datos - Nivel 2 - Gestión de órdenes

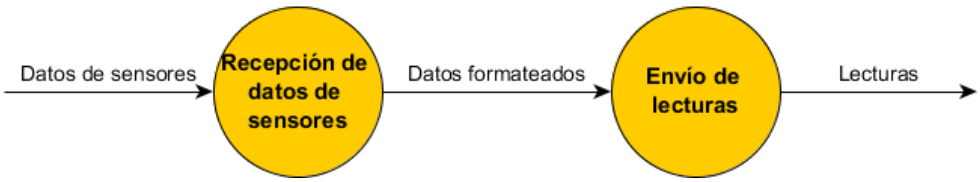


Ilustración 11 - Diagrama del Flujo de Datos - Nivel 2 - Procesado de datos

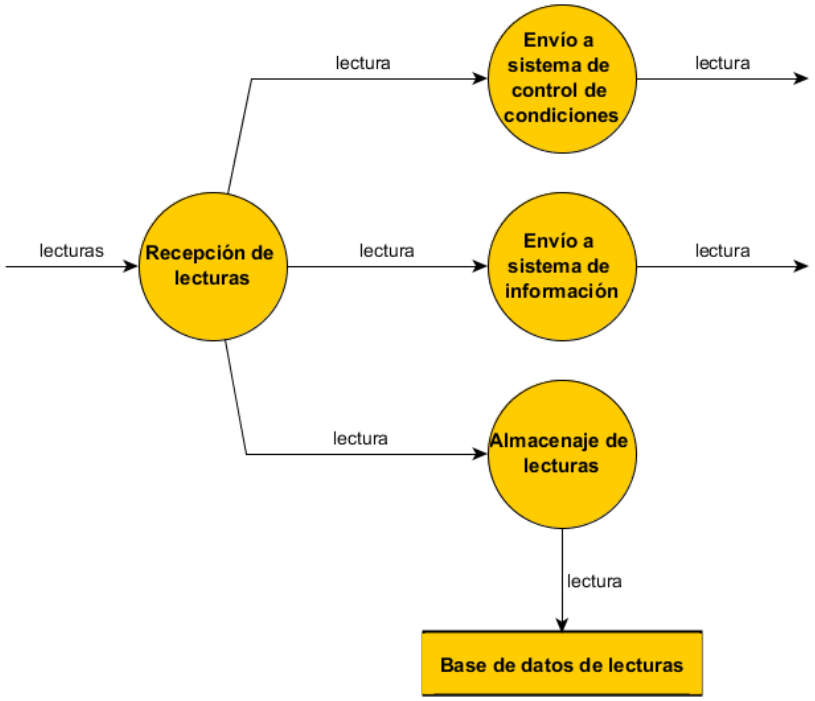


Ilustración 12 - Diagrama del Flujo de Datos - Nivel 2 - Gestión de lecturas

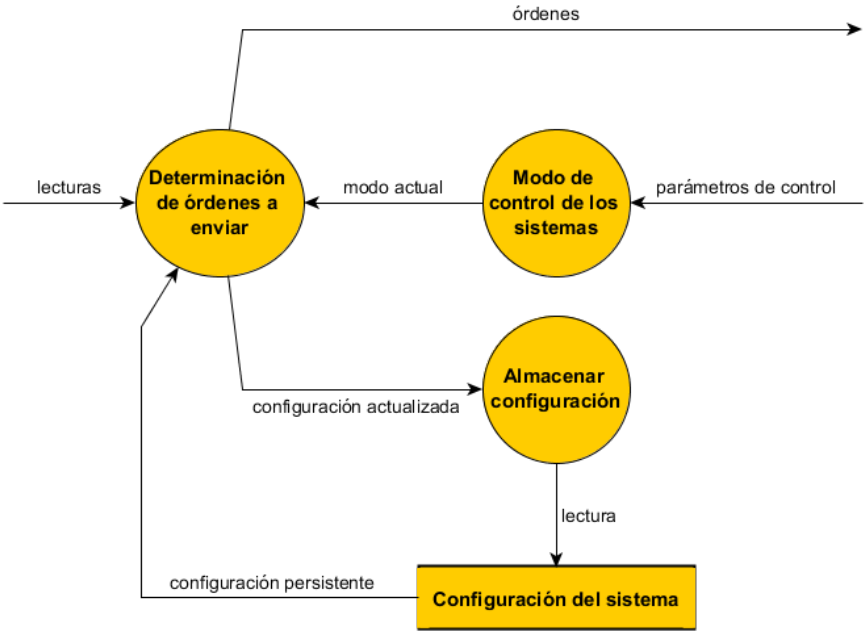


Ilustración 13 - Diagrama del Flujo de Datos - Nivel 2 – Control de condiciones

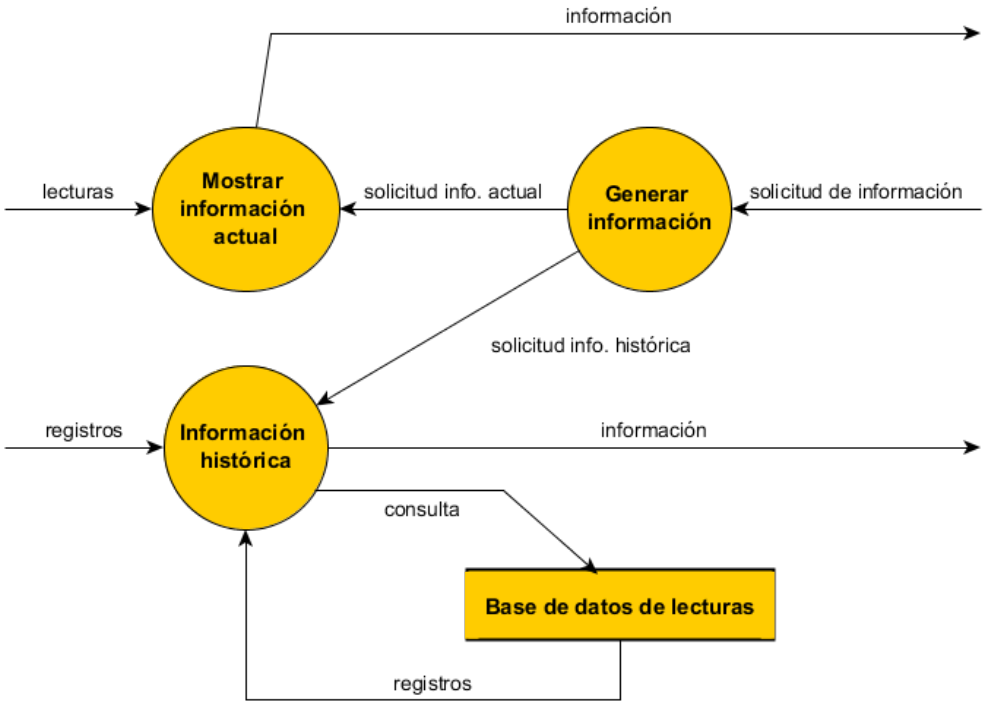


Ilustración 14 - Diagrama del Flujo de Datos - Nivel 2 - Sistema de información

Se ha optado por desarrollar los Diagramas de Flujo de Datos hasta el nivel 2 para cada nodo mostrado en el DFD-Nivel 1, pues este nivel ofrece una buena visión del intercambio de información en forma bastante concisa pero sin entrar en el nivel de detalle de niveles siguientes que supondrían explotar cada nodo de nivel 2 a un árbol demasiado grande y complejo.

2.2 MODELO DE DATOS DEL SISTEMA

El modelo de datos del sistema tiene como objetivo describir las estructuras de datos que existen en el sistema y la forma en que se relacionan, así como las operaciones que se pueden efectuar sobre los datos.

Para describir estas estructuras se emplean Diagramas de Entidad-Relación (DER) con los que se representan gráficamente las entidades relevantes del sistema de información junto a las interrelaciones y propiedades correspondientes. En la siguiente ilustración aparece representado el diagrama Entidad-Relación del sistema Hortduino.

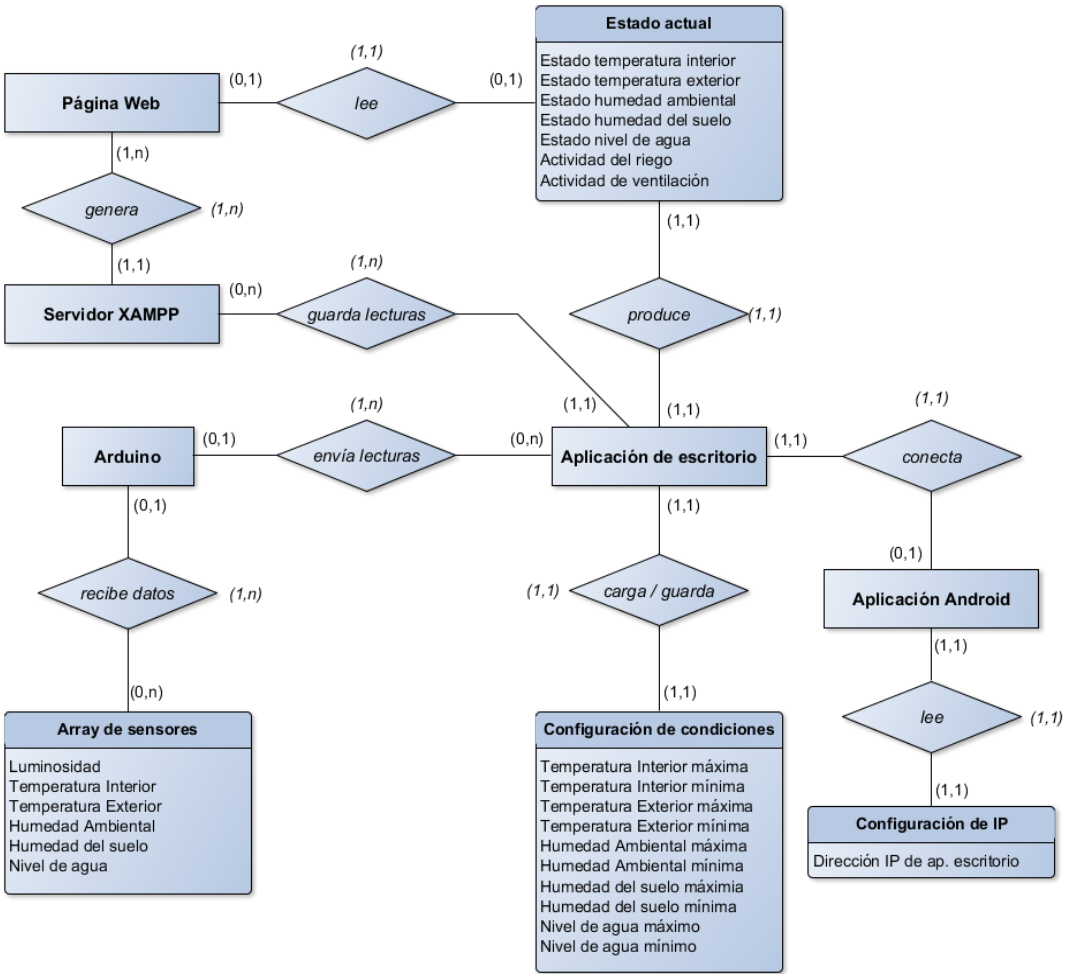


Ilustración 15 - Diagrama Entidad-Relación del sistema

En el diagrama de la ilustración anterior se han obviado algunos atributos de poca importancia para el sistema de datos como pueden ser direcciones IP y puertos con objeto de simplificar y facilitar la lectura del diagrama Entidad-Relación.

En este proyecto se ha optado por una aproximación mixta en lo referente al almacenamiento persistente de los diferentes datos usados por el sistema: mientras que las lecturas enviadas desde los sensores son almacenadas por la aplicación de escritorio en una base de datos para facilitar su posterior tratamiento, las configuraciones y estado actual del invernadero son almacenados en sendos archivos de texto, requeridos para el inicio y configuración de las aplicaciones de escritorio y Android.

3 DISEÑO DEL SISTEMA

3.1 DISEÑO DE LA BASE DE DATOS

Con la información disponible en el diagrama Entidad-Relación, se procede al diseño de la base de datos destinada a almacenar la información enviada desde el dispositivo Arduino. Para cada variable, se tiene en cuenta el rango máximo que pueden alcanzar los valores que ofrece cada sensor para asignar el tipo de campo correspondiente. Dado que únicamente se considera la entidad *Array de sensores* como objeto de almacenaje, será necesaria una única tabla para pasar del modelo Entidad-Relación al modelo relacional.

Además de la información procedente de las lecturas, la base de datos utilizará dos campos extra, añadidos para posibilitar consultas mas específicas: un índice, que se asignará de manera unívoca a cada registro añadido a la base de datos y será la clave primaria para acceder a la tabla, y la marca de tiempo (*time stamp*) en la que se incorpora el nuevo registro a la base de datos.

A partir de lo anterior, la estructura de las tablas de la base de datos se ha establecido de la siguiente forma:

Tablas *lecturas* y *actual*

Columna	Tipo	Nulo	Predeterminado	Enlaces a	Comentarios
Indice	int(11)	No	-	lecturas → Indice	Único, autoincremento
Fecha	timestamp	No	CURRENT_TIMESTAMP		
Luminosidad	int(4)	No	-		
TempExterior	float	No	-		
TempInterior	int(3)	No	-		
HumAmbiente	int(3)	No	-		
HumSuelo	int(4)	No	-		
NivelAgua	int(4)	No	-		
EstadoRiego	int(1)	No	-		
EstadoVent	int(1)	No	-		

Tabla 68 - Estructura interna de la base de datos

La tabla *actual* de la base de datos es la encargada de almacenar la lectura mas reciente enviada por Arduino. Posee una estructura idéntica a la tabla *lecturas,* en la que se almacenan todas las lecturas recibidas, pero con la peculiaridad de que sólo es usado un registro que se actualiza con la nueva información recibida de forma mas regular que en la tabla *lecturas*.

3.2 DISEÑO DE LOS ARCHIVOS DE CONFIGURACIÓN

Los archivos de configuración son los encargados de almacenar la configuración de los rangos aceptables para las condiciones supervisadas y el modo de control de los sistemas de riego y ventilación, todo ello basado en la elección del usuario. El formato usado en cada uno de ellos es el de valores numéricos separados mediante un caracter delimitador (coma o punto y coma), todo ello en una única línea de texto plano. El orden específico en el que se aparecen los valores se asocia a cada variable mediante el siguiente convenio:

Archivo *config.txt*

<luminosidad>;<temperatura interior>;<temperatura exterior>;<humedad ambiental>;<humedad del suelo>;<nivel de agua>;<modo de control sistema riego>;<modo de control sistema ventilación>;

El código asociado a los modos de control de los sistemas de riego y ventilación establece las siguientes convenciones:

- 0 : El sistema se establece en modo apagado (OFF) manualmente por el usuario
- 1 : El sistema se establece en modo encendido (ON) manualmente por el usuario
- 2 : El sistema se establece en modo de funcionamiento automático (AUTO) y actúa en base a la configuración aceptable de rangos que el usuario ha establecido.

Archivo *condiciones.txt*

<estado temperatura interior>,<estado temperatura exterior>,<estado humedad ambiental>,<estado humedad del suelo>,<estado nivel de agua>,<actividad sistema de riego>,<actividad sistema de ventilación>

El código interno para representar cada estado se establece en lo siguiente:

- -1 : El estado actual indica que no se alcanza el valor mínimo establecido para esa variable
- 0 : El estado actual indica una situación normal (dentro de los rangos permitidos)
- 1 : El estado actual indica que se ha sobrepasado el límite máximo fijado para esa variable

Para indicar la actividad de los sistemas de riego y ventilación se utiliza el valor 1 si el sistema en cuestión se encuentra activo en ese momento y el valor 0 en caso contrario.

Archivo *ip.txt*

Este archivo contiene la dirección IP de la aplicación de escritorio. Es necesario para indicar a la aplicación Android la dirección a la cual tiene que solicitar el reenvío de las lecturas y se genera mediante la invocación del script PHP *getip.php* que escribe en el archivo una línea de texto con formato de dirección IP (cuatro grupos de tres dígitos, separados por puntos)

3.3 DISEÑO DE LAS APLICACIONES

3.3.1 DIAGRAMA DE PAQUETES

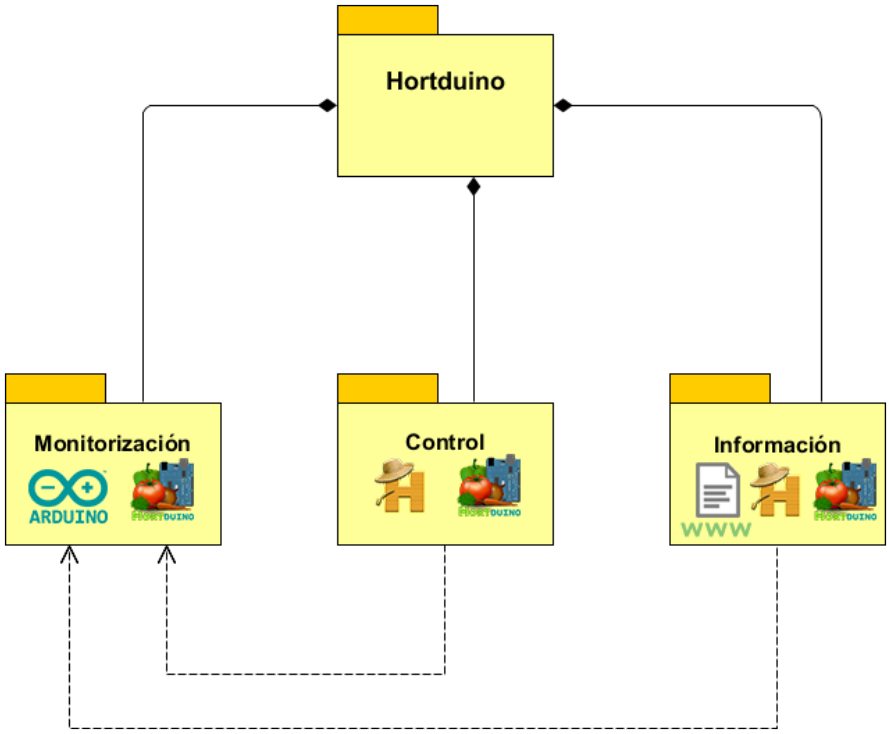


Ilustración 16 - Diagrama de paquetes

En la anterior ilustración se puede ver la división en paquetes del sistema Hortduino y sus dependencias. Los paquetes representan las capacidades disponibles en el sistema y muestran en su interior las herramientas encargadas de las funciones en cada caso.

Para las labores de monitorización, la aplicación de escritorio es la encargada de las funciones de supervisión de las condiciones del invernadero, recibidas en las lecturas que el dispositivo Arduino envía a dicha aplicación de forma periódica.

Las funciones de control pueden ser ejecutadas por la aplicación de escritorio o bien por la aplicación para dispositivos Android actuando a través de la aplicación de escritorio.

Por último, las tareas sobre manejo de información pueden ser efectuadas por las aplicaciones de escritorio y Android (para visualizar datos en tiempo real y guardar la información recibida en la base de datos), y por la página web si se desea consultar la información almacenada en la base de datos. La aplicación de escritorio es la pieza fundamental del sistema, ya que interviene en las tres funciones básicas que presenta el sistema Hortduino.

3.3.2 DIAGRAMA DE FLUJO DEL DISPOSITIVO ARDUINO

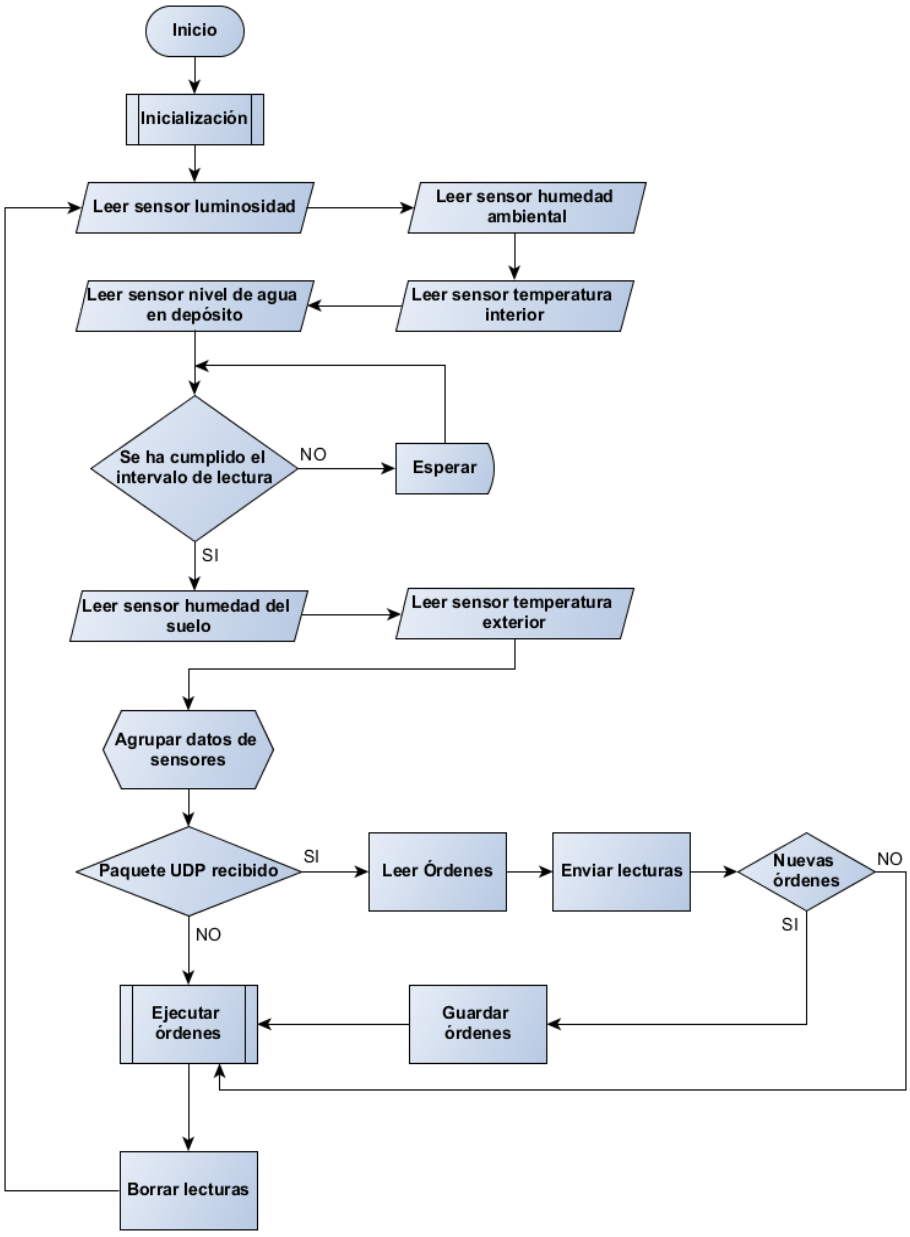


Ilustración 17: Diagrama de flujo del dispositivo Arduino

3.3.3 DIAGRAMA DE FLUJO DE LA APLICACIÓN DE ESCRITORIO

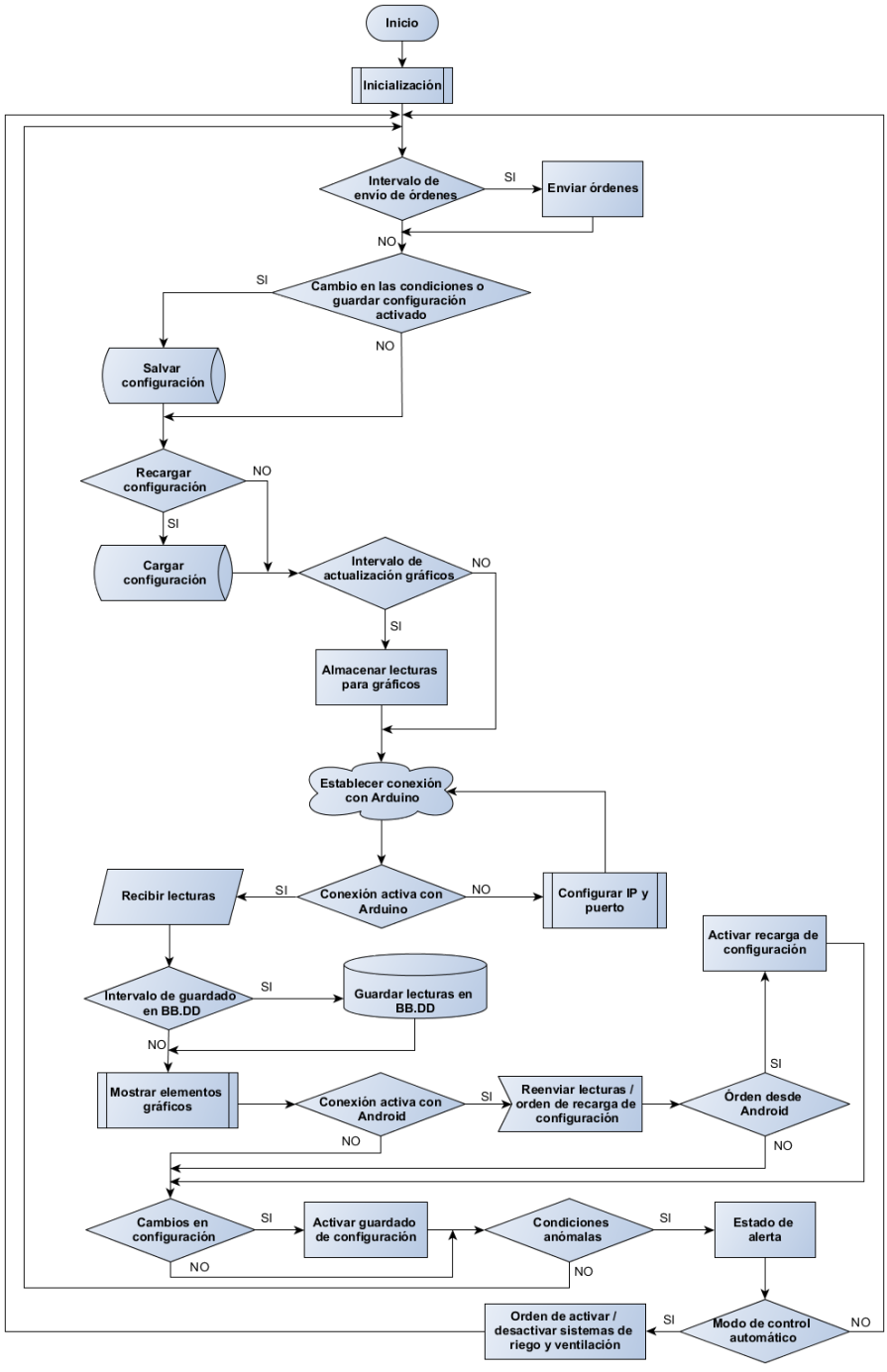


Ilustración 18 - Diagrama de flujo de la aplicación de escritorio

3.3.4 DIAGRAMA DE FLUJO DE LA APLICACIÓN ANDROID

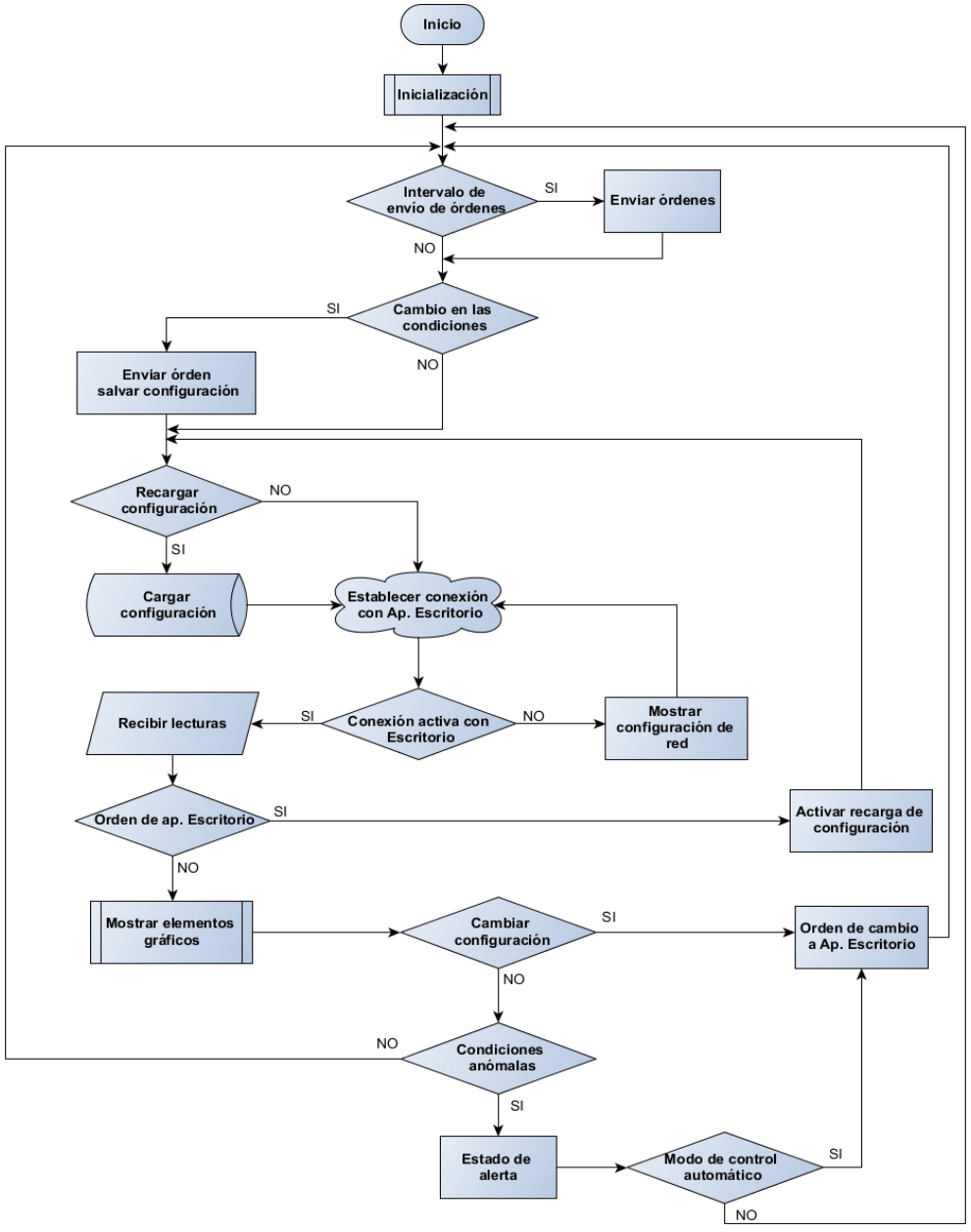


Ilustración 19 - Diagrama de flujo de la aplicación Android

3.3.5 DIAGRAMA DE FLUJO DE LA PÁGINA WEB

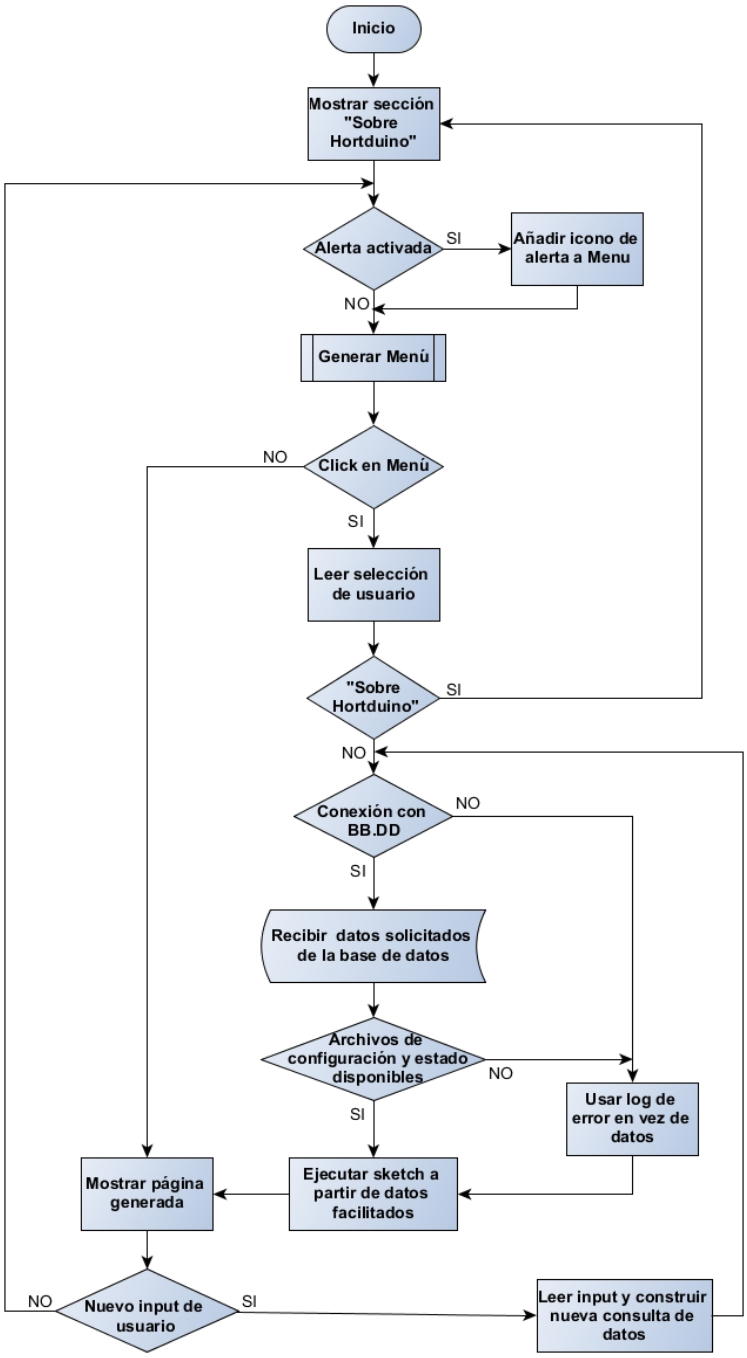


Ilustración 20 - Diagrama de flujo de la página web

3.3.6 PROCEDIMIENTOS DE LA APLICACIÓN ARDUINO

ID	**setup**
Parámetros	-
Descripción	Procedimiento de inicialización que es ejecutado automáticamente cuando se inicia el dispositivo Arduino. Solo puede ser ejecutado una vez.
Procedimientos asociados	loop

Tabla 1 - Procedimiento setup

ID	**loop**
Parámetros	-
Descripción	Bucle del procedimiento principal. Es invocado después de ejecutar el procedimiento setup
Procedimientos asociados	setup lecturaSensores recibirOrdenes ejecutarOrdenes

Tabla 2 - Procedimiento loop

ID	**lecturaSensores**		
Parámetros	**Nombre**	**E/S**	**Función**
	pinLDR	entrada	pin de entrada analógica del fotorresistor
	pinLM35	entrada	pin de entrada analógica del sensor de temperatura exterior
	pinYeso	entrada	pin de entrada analógica del sensor de humedad del suelo
	pinDHT	entrada	pin de entrada analógica del sensor de humedad ambiental y temperatura interior
	tipoDHT	entrada	tipo de sensor de la familia DHT
	datos	salida	cadena de texto con las lecturas formateadas de los sensores
Descripción	Genera una lectura con los valores recibidos en los pines indicados en el intervalo fijado.		
Procedimientos asociados	loop lecturaPing		

Tabla 3 - Procedimiento lecturaSensores

ID	**lecturaPing**
Parámetros	-
Descripción	Cálculo porcentual de nivel de agua en el depósito de riego mediante envío y recepción de pulsos ultrasónicos. El valor devuelto es usado en el procedimiento lecturaSensores para componer la lectura.
Procedimientos asociados	lecturaSensores

Tabla 4 - Procedimiento lecturaPing

ID	recibirOrdenes		
Parámetros	Nombre	E/S	Función
	ordenes	salida	Órdenes recibidas en formato numérico (3 bytes)
Descripción	Extrae las órdenes a ejecutar del paquete UDP, si éste está disponible en el buffer de entrada. Si no se ha recibido dicho paquete UDP, las órdenes anteriores son mantenidas.		
Procedimientos asociados	loop		

Tabla 5 - Procedimiento lecturaSensores

ID	enviarDatos		
Parámetros	Nombre	E/S	Función
	datos	entrada	Datos formateados de la lectura de sensores
Descripción	Crea un paquete UDP conteniendo los datos formateados de la lectura de sensores y lo envía a la dirección IP que lo solicitó. Después del envío, los datos son borrados.		
Procedimientos asociados	ejecutarOrdenes		

Tabla 6 - Procedimiento enviarDatos

ID	ejecutarOrdenes		
Parámetros	Nombre	E/S	Función
	ordenes	entrada	Ordenes a ejecutar en los sistemas de riego y ventilación
Descripción	Si se han recibido órdenes, este procedimiento cambia el estado de los sistemas de riego y ventilación basándose en el código correspondiente dentro del paquete de órdenes e invoca al procedimiento de envío de datos.		
Procedimientos asociados	loop enviarDatos estadoRiego estadoVentilación		

Tabla 7 - Procedimiento ejecutarOrdenes

ID	estadoRiego		
Parámetros	Nombre	E/S	Función
	ordenes	entrada	Ordenes a ejecutar en los sistemas de riego y ventilación
Descripción	Activa o desactiva el sistema de riego basándose en la orden recibida		
Procedimientos asociados	ejecutarOrdenes		

Tabla 8 - Procedimiento estadoRiego

ID	estadoVent		
Parámetros	Nombre	E/S	Función
	ordenes	entrada	Ordenes a ejecutar en los sistemas de riego y ventilación
Descripción	Activa o desactiva el sistema de ventilación basándose en la orden recibida		
Procedimientos asociados	ejecutarOrdenes		

Tabla 9 - Procedimiento estadoVent

3.3.7 PROCEDIMIENTOS DE LA APLICACIÓN DE ESCRITORIO

ID	**setup**
Parámetros	-
Descripción	Procedimiento de inicialización que es ejecutado automáticamente cuando se inicia la aplicación de escritorio. Inicializa variables y conexiones.
Procedimientos asociados	inicializarAnimacion inicializaConfig draw

Tabla 10 - Procedimiento setup

ID	**draw**
Parámetros	-
Descripción	Bucle del programa principal. Es invocado automáticamente después del procedimiento setup y se encarga de establecer los intervalos para recepción de lecturas, envío de órdenes, guardado de datos y mostrar los elementos gráficos de la interfaz.
Procedimientos asociados	enviarOrdenes salvarConfig inicializaConfig establecerUltimaAct establecerConexion dibujarBotonesyEstados dibujarGraficosyEjes dibujarSelectoresAct mostrarDatos almacenarLecturas comprobarCondiciones salvarLecturas

Tabla 11 - Procedimiento draw

ID	**establecerConexion**
Parámetros	-
Descripción	Determina si la conexión con el dispositivo Arduino está activa en función de la recepción de paquetes UDP en un intervalo de tiempo fijado. Si no se recibe el número de paquetes esperados, se marca la conexión como inactiva y se invoca el procedimiento que muestra la ventana de configuración de red.
Procedimientos asociados	draw

Tabla 12 - Procedimiento establecerConexion

ID	**mostrarMenuConfigConex**
Parámetros	-
Descripción	Muestra el menú de configuración de red en el caso de que no se reciban paquetes UDP desde Arduino en un intervalo fijado. El usuario puede cambiar la IP y puerto a la que intentar conectar.
Procedimientos asociados	establecerConexion

Tabla 13 - Procedimiento mostrarMenuConfigConex

ID	inicializaConfig		
Parámetros	**Nombre**	**E/S**	**Función**
	arrayCondiciones	salida	Condiciones máximas y mínimas fijadas por el usuario para las variables
	estadoOrdenes	salida	Modo de control de los sistemas de riego y ventilación
Descripción	Carga la configuración almacenada en el archivo *config.txt* establecida por el usuario.		
Procedimientos asociados	setup draw		

Tabla 14 - Procedimiento inicializaConfig

ID	inicializarAnimacion		
Parámetros	**Nombre**	**E/S**	**Función**
	anRiego	salida	Frames de la animación de riego activo
	anVent	salida	Frames de la animación de ventilación activa
Descripción	Carga en memoria las animaciones del riego y ventilación.		
Procedimientos asociados	setup		

Tabla 15 - Procedimiento inicializarAnimacion

ID	almacenarLecturas		
Parámetros	**Nombre**	**E/S**	**Función**
	arrayDatos	salida	Array con los datos recibidos de los sensores
	indice	salida	Posición de inserción en los arrays de gráficos temporales
	lum	salida	Array de valores de luminosidad
	tempLM35	salida	Array de valores de temperatura exterior
	tempDHT	salida	Array de valores de temperatura interior
	humDTH	salida	Array de valores de humedad ambiental
	humSuelo	salida	Array de valores de humedad del suelo
	nivelH2O	salida	Array de valores del nivel de agua en depósito
Descripción	Guarda en los arrays utilizados para mostrar los gráficos temporales las lecturas recibidas si se ha cumplido el intervalo de actualización de gráficos. Cada lectura se guarda en la posición indicada por el parámetro *indice*. En caso de que *indice* alcance el tamaño de actualizaciones de pantalla fijado, es reiniciado a 0.		
Procedimientos asociados	draw		

Tabla 16 - Procedimiento almacenarLecturas

ID	salvarLecturas		
Parámetros	Nombre	E/S	Función
	datos	entrada	Datos de la última lectura de los sensores recibida
Descripción	Guarda la última lectura recibida en la base de datos del servidor en el intervalo fijado para ello mediante una consulta en SQL.		
Procedimientos asociados	draw		

Tabla 17 - Procedimiento salvarLecturas

ID	salvarConfig
Parámetros	-
Descripción	Guarda la configuración sobre los rangos aceptables para las variables y el modo de control de cada sistema en el archivo *config.txt* del servidor. Si la conexión con la aplicación Android está activa y se necesita salvar la configuración, envía un mensaje hacia la aplicación Android para notificar este evento.
Procedimientos asociados	draw receive

Tabla 18 - Procedimiento salvarConfig

ID	mostrarDatos		
Parámetros	Nombre	E/S	Función
	arrayDatos	entrada	Datos de la última lectura de los sensores recibida
Descripción	Muestra en pantalla los nombres de las variables y el último valor recibido para cada una.		
Procedimientos asociados	draw		

Tabla 19 - Procedimiento mostrarDatos

ID	establecerUltimaAct		
Parámetros	Nombre	E/S	Función
	ultimaAct	salida	fecha y hora de la recepción de la última lectura de los sensores
Descripción	Muestra en pantalla la fecha y hora de la recepción de la última lectura de los sensores.		
Procedimientos asociados	draw		

Tabla 20 - Procedimiento establecerUltimaAct

ID	comprobarCondiciones		
Parámetros	**Nombre**	**E/S**	**Función**
	arrayCondiciones	entrada	Array con los rangos aceptables de las variables
	estCondiciones	salida	Estado de las condiciones del invernadero en código numérico
Descripción	Comprueba si el valor actual de las condiciones del invernadero se encuentra dentro de los rangos aceptables fijados por el usuario, mostrando en pantalla el estado de las condiciones mediante líneas de texto informativas junto a un icono representativo del estado actual. Devuelve una cadena con el código del estado de las condiciones.		
Procedimientos asociados	draw		

Tabla 21 - Procedimiento comprobarCondiciones

ID	dibujarSelectorActualizacion		
Parámetros	**Nombre**	**E/S**	**Función**
	xx	entrada	Posición de la esquina superior izq. del selector en el eje X
	y	entrada	Posición de la esquina superior izq. del selector en el eje Y
	ancho	entrada	Ancho del selector
	alto	entrada	Alto del selector
	valor	entrada	Valor numérico a mostrar en la caja del selector
	texto	entrada	Texto que acompaña al selector
	id	entrada	Identificador único del selector
Descripción	Dibuja en la pantalla el selector de horas, minutos o segundos según corresponda y los controles para modificar el intervalo de actualización de los gráficos en pantalla.		
Procedimientos asociados	dibujarSelectoresAct		

Tabla 22 - Procedimiento dibujarSelectorActualización

ID	dibujarSelectoresAct		
Parámetros	**Nombre**	**E/S**	**Función**
	x	entrada	Posición de la esquina superior izq. del selector en el eje X
	y	entrada	Posición de la esquina superior izq. del selector en el eje Y
	intervaloAct	salida	Intervalo de actualización de los gráficos en pantalla
Descripción	Dibuja en la pantalla los selectores para modificar el intervalo de actualización de los gráficos en pantalla. Si el usuario modifica el intervalo de actualización, devuelve el nuevo intervalo.		
Procedimientos asociados	draw dibujarSelectoresAct		

Tabla 23 - Procedimiento dibujarSelectoresAct

ID	dibujarBotonesyEstados
Parámetros	-
Descripción	Dibuja en la pantalla los elementos gráficos correspondientes a los botones y estados de los sistemas de riego y ventilación y los selectores de los rangos aceptables para las variables.
Procedimientos asociados	draw dibujarBoton dibujarSelectores dibujarEstadoOrdenes

Tabla 24 - Procedimiento dibujarBotonesyEstados

ID	dibujarGraficosyEjes
Parámetros	-
Descripción	Dibuja en la pantalla los gráficos temporales de la luminosidad, temperatura interior y exterior, humedad ambiental y humedad del suelo con sus correspondientes ejes. También muestra gráficamente el porcentaje de agua restante en el depósito de riego.
Procedimientos asociados	draw dibujarGrafico dibujarEjes dibujarDeposito

Tabla 25 - Procedimiento dibujarGraficosyEjes

ID	dibujarSelector		
Parámetros	**Nombre**	**E/S**	**Función**
	x	entrada	Posición de la esquina superior izq. del selector en el eje X
	y	entrada	Posición de la esquina superior izq. del selector en el eje Y
	texto	entrada	Intervalo de actualización de los gráficos en pantalla
	valor	entrada	Valor actual del selector
	id	entrada	Identificador único del selector
	maximo	salida	Valor máximo permisible
	mínimo	salida	Valor mínimo permisible
Descripción	Dibuja en la pantalla el selector correspondiente al rango máximo o mínimo de la variable indicada junto a los controles para su modificación, permitiendo al usuario especificar el valor admisible. Si el usuario cambia el valor mostrado, se informa del cambio de condiciones al procedimiento *salvarConfig()*.		
Procedimientos asociados	dibujarSelectores		

Tabla 26 - Procedimiento dibujarSelector

ID	dibujarSelectores		
Parámetros	**Nombre**	**E/S**	**Función**
	x	entrada	Posición de la esquina superior izq. del grupo en el eje X
	y	entrada	Posición de la esquina superior izq. del grupo en el eje Y
	arrayCondiciones	salida	Array con los rangos aceptables de las variables
Descripción	Dibuja en la pantalla el conjunto de selectores de rango para las variables monitorizadas.		
Procedimientos asociados	dibujarSelector		

Tabla 27 - Procedimiento dibujarSelectores

ID	dibujarEstadoOrdenes		
Parámetros	**Nombre**	**E/S**	**Función**
	x	entrada	Coordenada de posición en el eje X
	y	entrada	Coordenada de posición en el eje Y
	estadoOrdenes	entrada	Modo de control de los sistemas de riego y ventilación
	animRiego	entrada	Array de frames de la animación de riego
	animVent	entrada	Array de frames de la animación de ventilación
Descripción	Muestra en pantalla la actividad de los sistemas de riego y ventilación mediante imágenes estáticas (inactivo) o animaciones (activo)		
Procedimientos asociados	dibujarBotonesyEstados		

Tabla 28 - Procedimiento dibujarEstadoOrdenes

ID	dibujarGrafico		
Parámetros	**Nombre**	**E/S**	**Función**
	datosSensor	entrada	Array de datos recibidos del sensor
	limInferior	entrada	Valor mínimo de lectura del sensor
	limSuperior	entrada	Valor máximo de lectura del sensor
	objInferior	entrada	Valor mínimo en el eje Y del gráfico
	objSuperior	entrada	Valor máximo en el eje Y del gráfico
	r	entrada	Valor del componente rojo del color del gráfico
	gg	entrada	Valor del componente verde del color del gráfico
	bb	entrada	Valor del componente azul del color del gráfico
	yPos	entrada	Posición de partida del gráfico en el eje Y
Descripción	Dibuja en la pantalla el gráfico de la variable en cuestión a partir a los valores contenidos en el array de datos correspondiente, empleando curvas construidas por mapeado de rangos límite y objetivo. El tamaño máximo del gráfico respecto al eje X viene determinado por el numero de actualizaciones en pantalla permitidas. Si se alcanza el número máximo de éstas, el gráfico se redibuja desde el origen del eje Y, manteniendo el trazado anterior y actualizando progresivamente la línea en la posición indicada por el punto blanco.		
Procedimientos asociados	dibujarGraficosyEjes		

Tabla 29 - Procedimiento dibujarGrafico

ID	dibujarDeposito		
Parámetros	**Nombre**	**E/S**	**Función**
	x	entrada	Coordenada en el eje X del borde superior izq. del depósito
	y	entrada	Coordenada en el eje Y del borde superior izq. del depósito
	ancho	entrada	Ancho del depósito en pixels
	alto	entrada	Alto del depósito en pixels
	nivel	entrada	Nivel porcentual de llenado del depósito
Descripción	Dibuja en la pantalla la representación gráfica del nivel de agua actual en el depósito para riego. Se indica el porcentaje de llenado encima de la representación del nivel actual de agua. La opacidad del color azul depende del valor del nivel (mayor transparencia a menor nivel de llenado).		
Procedimientos asociados	dibujarGraficosyEjes		

Tabla 30 - Procedimiento dibujarDeposito

ID	dibujarEjes		
Parámetros	**Nombre**	**E/S**	**Función**
	xPos	entrada	Coordenada en el eje X del origen del eje X del gráfico
	yPos	entrada	Coordenada en el eje Y del origen del eje Y del gráfico
	ancho	entrada	Ancho del eje en pixels
	alto	entrada	Alto del eje en pixels
	leyenda	entrada	Texto que se muestra debajo del gráfico
Descripción	Dibuja en la pantalla los ejes X e Y que acompañan a los gráficos de las variables		
Procedimientos asociados	dibujarGraficosyEjes mostrarLeyenda		

Tabla 31 - Procedimiento dibujarEjes

ID	mostrarLeyenda		
Parámetros	**Nombre**	**E/S**	**Función**
	x	entrada	Coordenada en el eje X del origen del eje X del gráfico
	y	entrada	Coordenada en el eje Y del origen del eje Y del gráfico
	ancho	entrada	Ancho del eje en pixels
	zoomEjeX	entrada	Separación entre valores del gráfico en el eje X
	texto	entrada	Texto que se muestra debajo del gráfico
Descripción	Muestra la leyenda del gráfico debajo del eje X centrada respecto al su ancho		
Procedimientos asociados	dibujarEjes		

Tabla 32 - Procedimiento mostrarLeyenda

ID	dibujarBoton		
Parámetros	**Nombre**	**E/S**	**Función**
	x	entrada	Coordenada en el eje X del botón
	y	entrada	Coordenada en el eje Y del botón
	ancho	entrada	Ancho del botón en pixels
	alto	entrada	Alto del botón en pixels
	estadoOrdenes	entrada	Modo de control de los sistemas de riego y ventilación
	id	entrada	Identificador del botón (riego o ventilación)
Descripción	Dibuja el botón para cambiar el modo de control de los sistemas de riego y ventilación. El modo de control es mostrado como texto en el interior del botón, el cual presenta un color diferente para cada modo (verde=ON, rojo=OFF, amarillo=AUTO)		
Procedimientos asociados	dibujarBotonesyEstados		

Tabla 33 - Procedimiento dibujarBoton

ID	punteroSobreBoton		
Parámetros	**Nombre**	**E/S**	**Función**
	x	entrada	Coordenada en el eje X
	y	entrada	Coordenada en el eje Y
	ancho	entrada	Ancho del botón en pixels
	alto	entrada	Alto del botón en pixels
Descripción	Devuelve si el puntero del ratón se encuentra dentro del área del botón		
Procedimientos asociados	mouseClicked		

Tabla 34 - Procedimiento punteroSobreBoton

ID	punteroSobreTrianguloArriba		
Parámetros	**Nombre**	**E/S**	**Función**
	x1	entrada	Coordenada en el eje X del punto 1
	y1	entrada	Coordenada en el eje Y del punto 1
	x2	entrada	Coordenada en el eje X del punto 2
	y2	entrada	Coordenada en el eje Y del punto 2
	x3	entrada	Coordenada en el eje X del punto 3
	y3	entrada	Coordenada en el eje Y del punto 3
Descripción	Devuelve si el puntero del ratón se encuentra dentro del área del triángulo superior del selector.		
Procedimientos asociados	dibujarSelectorActualizacion dibujarSelector		

Tabla 35 - Procedimiento punteroSobreTrianguloArriba

ID	punteroSobreTrianguloAbajo		
Parámetros	**Nombre**	**E/S**	**Función**
	x1	entrada	Coordenada en el eje X del punto 1
	y1	entrada	Coordenada en el eje Y del punto 1
	x2	entrada	Coordenada en el eje X del punto 2
	y2	entrada	Coordenada en el eje Y del punto 2
	x3	entrada	Coordenada en el eje X del punto 3
	y3	entrada	Coordenada en el eje Y del punto 3
Descripción	Devuelve si el puntero del ratón se encuentra dentro del área del triángulo inferior del selector.		
Procedimientos asociados	dibujarSelectorActualizacion dibujarSelector		

Tabla 36 - Procedimiento punteroSobreTrianguloAbajo

ID	enviarOrdenes		
Parámetros	**Nombre**	**E/S**	**Función**
	ordenes	entrada	Contenido a enviar en el paquete UDP
Descripción	Si ha transcurrido el intervalo fijado para el envío de órdenes, este procedimiento construye el paquete UDP a enviar conteniendo las órdenes a ejecutar en Arduino. Dichas órdenes dependerán del modo de control activo y de las condiciones de ese momento. Después de su envío, las órdenes son borradas.		
Procedimientos asociados	draw		

Tabla 37 - Procedimiento enviarOrdenes

ID	mouseClicked
Parámetros	-
Descripción	Cambia el estado del modo de control de los sistemas de riego y ventilación si el usuario hace click con el ratón encima de los botones correspondientes.
Procedimientos asociados	draw punteroSobreBoton

Tabla 38 - Procedimiento mouseClicked

ID	mouseReleased
Parámetros	-
Descripción	Informa si el botón del ratón no está presionado
Procedimientos asociados	draw

Tabla 39 - Procedimiento mouseReleased

ID	**keyReleased**
Parámetros	-
Descripción	Almacena la última tecla pulsada en el teclado.
Procedimientos asociados	draw

Tabla 40 - Procedimiento keyReleased

ID	**receive**		
Parámetros	**Nombre**	**E/S**	**Función**
	mensaje	entrada	Contenido del paquete UDP
	IPAdress	entrada	Dirección IP del dispositivo Arduino
	port	entrada	Puerto del dispositivo Arduino
Descripción	Este procedimiento es el handler de la librería UDP sobre eventos de recepción de datagramas. Se han incluido varias funcionalidades extra dentro de este procedimiento para ampliar su utilidad: • Comprueba si la dirección que envió el paquete UDP se corresponde con la aplicación Android. Si esto se cumple, se examina el paquete para determinar si contiene la orden de recargar la configuración desde el archivo *config.txt* (modificado por la aplicación Android previamente) para mantener la consistencia de configuración entre las aplicaciones o simplemente es una petición de reenvío de lecturas. • Si la dirección remitente del paquete UDP es la del dispositivo Arduino, el mensaje se divide mediante el caracter separador "," y los valores son almacenados en las posiciones correspondientes a las variables en un array. • Cada vez que se registra un evento de recepción, el contador de paquetes esperados empleado en el procedimiento *establecerConexion()* se inicializa a 0.		
Procedimientos asociados	draw		

Tabla 41 - Procedimiento receive

3.3.8 PROCEDIMIENTOS DE LA APLICACIÓN ANDROID

ID	**setup**
Parámetros	-
Descripción	Procedimiento de inicialización que es ejecutado automáticamente cuando se inicia la aplicación de escritorio. Inicializa variables y conexiones.
Procedimientos asociados	draw inicializaConfig inicializarAnimacion

Tabla 42 - Procedimiento setup

ID	**draw**
Parámetros	-
Descripción	Bucle del programa principal. Es invocado automáticamente después del procedimiento setup y se encarga de establecer los intervalos para recepción de lecturas, envío de órdenes y mostrar los elementos gráficos de la interfaz.
Procedimientos asociados	enviarOrdenes salvarConfig inicializaConfig establecerConexion mostrarInicio mostrarMenu dibujarDeposito dibujarSelectores mostrarMenuInfo

Tabla 43 - Procedimiento draw

ID	**dispatchTouchEvent**		
Parámetros	**Nombre**	**E/S**	**Función**
	event	entrada	Evento táctil
Descripción	Redefinición del método que gestiona los eventos táctiles en Android para simular las funciones del ratón en Processing.		
Procedimientos asociados	draw		

Tabla 44 - Procedimiento dispatchTouchEvent

ID	mostrarInicio
Parámetros	-
Descripción	Muestra la pantalla principal de la aplicación con la información recibida desde la aplicación de escritorio, los controles y estados de los sistemas de riego y ventilación y el botón de acceso al menú.
Procedimientos asociados	draw mostrarDatos comprobarCondiciones dibujarEstadoOrdenes

Tabla 45 - Procedimiento mostrarInicio

ID	mostrarMenu
Parámetros	-
Descripción	Muestra el menú de opciones en forma de varios botones.
Procedimientos asociados	draw dibujarBotonesMenu

Tabla 46 - Procedimiento mostrarMenu

ID	mostrarMenuInfo
Parámetros	-
Descripción	Muestra la dirección IP a la que se conecta y el número de actualizaciones recibidas. Solo se usa a modo de comprobación.
Procedimientos asociados	draw dibujarBotonesMenu

Tabla 47 - Procedimiento mostrarMenuInfo

ID	dibujarBotonesMenu
Parámetros	-
Descripción	Presenta en pantalla los botones correspondientes a cada opción del menú y controla la opción elegida por el usuario.
Procedimientos asociados	mostrarMenu

Tabla 48 - Procedimiento dibujarBotonesMenu

ID	dibujarSelectores
Parámetros	-
Descripción	Muestra la pantalla de configuración de los rangos máximos y mínimos para cada variable del invernadero. Si el usuario modifica alguno de estos valores, se activa el guardado de la nueva configuración en el servidor a través de la aplicación de escritorio.
Procedimientos asociados	draw

Tabla 49 - Procedimiento dibujarSelectores

ID	establecerConexion
Parámetros	-
Descripción	Si no se han recibido paquetes UDP desde la aplicación de escritorio en el tiempo estipulado, muestra la información actual de la conexión (IP y puerto)
Procedimientos asociados	draw

Tabla 50 - Procedimiento establecerConexion

ID	inicializaConfig		
Parámetros	Nombre	E/S	Función
	event	entrada	Evento táctil
Descripción	Redefinición del método que gestiona los eventos táctiles en Android para simular las funciones del ratón en Processing.		
Procedimientos asociados	draw		

Tabla 51 - Procedimiento dispatchTouchEvent

ID	inicializaConfig		
Parámetros	Nombre	E/S	Función
	arrayCondiciones	salida	Condiciones máximas y mínimas fijadas por el usuario para las variables
	estadoOrdenes	salida	Modo de control de los sistemas de riego y ventilación
Descripción	Carga la configuración almacenada en el archivo *config.txt* establecida por el usuario.		
Procedimientos asociados	setup draw		

Tabla 52 - Procedimiento inicializaConfig

ID	inicializarAnimacion		
Parámetros	Nombre	E/S	Función
	anRiego	salida	Frames de la animación de riego activo
	anVent	salida	Frames de la animación de ventilación activa
Descripción	Carga en memoria las animaciones del riego y ventilación.		
Procedimientos asociados	setup		

Tabla 53 - Procedimiento inicializarAnimacion

ID	salvarConfig
Parámetros	-
Descripción	Guarda la configuración sobre los rangos aceptables para las variables y el modo de control de cada sistema en el archivo *config.txt* del servidor. Si la conexión con la aplicación Android está activa y se necesita salvar la configuración, envía un mensaje hacia la aplicación Android para notificar este evento.
Procedimientos asociados	draw

Tabla 54 - Procedimiento salvarConfig

ID	mostrarDatos		
Parámetros	Nombre	E/S	Función
	arrayDatos	entrada	Datos de la última lectura de los sensores recibida
Descripción	Muestra en pantalla los nombres de las variables y el último valor recibido para cada una.		
Procedimientos asociados	mostrarInicio		

Tabla 55 - Procedimiento mostrarDatos

ID	comprobarCondiciones		
Parámetros	Nombre	E/S	Función
	arrayCondiciones	entrada	Array con los rangos aceptables de las variables
	estCondiciones	salida	Estado de las condiciones del invernadero en código numérico
Descripción	Comprueba si el valor actual de las condiciones del invernadero se encuentra dentro de los rangos aceptables fijados por el usuario, mostrando en pantalla el estado de las condiciones mediante líneas de texto informativas junto a un icono representativo del estado actual. Devuelve una cadena con el código del estado de las condiciones.		
Procedimientos asociados	mostrarInicio		

Tabla 56 - Procedimiento comprobarCondiciones

ID	dibujarEstadoOrdenes		
Parámetros	Nombre	E/S	Función
	x	entrada	Coordenada de posición en el eje X
	y	entrada	Coordenada de posición en el eje Y
	estadoOrdenes	entrada	Modo de control de los sistemas de riego y ventilación
	animRiego	entrada	Array de frames de la animación de riego
	animVent	entrada	Array de frames de la animación de ventilación
Descripción	Muestra en pantalla la actividad de los sistemas de riego y ventilación mediante imágenes estáticas (inactivo) o animaciones (activo)		
Procedimientos asociados	mostrarInicio		

Tabla 57 - Procedimiento dibujarEstadoOrdenes

ID	dibujarDeposito		
Parámetros	**Nombre**	**E/S**	**Función**
	x	entrada	Coordenada en el eje X del borde superior izq. del depósito
	y	entrada	Coordenada en el eje Y del borde superior izq. del depósito
	ancho	entrada	Ancho del depósito en pixels
	alto	entrada	Alto del depósito en pixels
	nivel	entrada	Nivel porcentual de llenado del depósito
Descripción	Dibuja en la pantalla la representación gráfica del nivel de agua actual en el depósito para riego. Se indica el porcentaje de llenado encima de la representación del nivel actual de agua. La opacidad del color azul depende del valor del nivel (mayor transparencia a menor nivel de llenado).		
Procedimientos asociados	draw		

Tabla 58 - Procedimiento dibujarDeposito

ID	dibujarBotonOrdenes		
Parámetros	**Nombre**	**E/S**	**Función**
	x	entrada	Coordenada en el eje X del borde superior izq. del depósito
	y	entrada	Coordenada en el eje Y del borde superior izq. del depósito
	ancho	entrada	Ancho del depósito en pixels
	alto	entrada	Alto del depósito en pixels
	estadoOrdenes	entrada	Modo de control de los sistemas de riego y ventilación
	id	entrada	Identificador del botón
Descripción	Muestra en la posición indicada de la pantalla el botón de cambio de modo correspondiente al sistema en cuestión. Si el usuario cambia de modo, el texto interior y el color del botón cambian para reflejar el estado actual		
Procedimientos asociados	mostrarInicio punteroSobreBoton		

Tabla 59 - Procedimiento dibujarBotonOrdenes

ID	enviarOrdenes		
Parámetros	**Nombre**	**E/S**	**Función**
	ordenes	entrada	Contenido a enviar en el paquete UDP
Descripción	Si ha transcurrido el intervalo fijado para el envío de órdenes, este procedimiento construye el paquete UDP a enviar conteniendo las órdenes a ejecutar en Arduino. Dichas órdenes dependerán del modo de control activo y de las condiciones de ese momento. Después de su envío a la aplicación de escritorio, las órdenes son borradas.		
Procedimientos asociados	draw		

Tabla 60 - Procedimiento enviarOrdenes

ID	punteroSobreBoton		
Parámetros	**Nombre**	**E/S**	**Función**
	x	entrada	Coordenada en el eje X
	y	entrada	Coordenada en el eje Y
	ancho	entrada	Ancho del botón en pixels
	alto	entrada	Alto del botón en pixels
Descripción	Devuelve si el dedo del usuario se encuentra dentro del área del botón		
Procedimientos asociados	dibujarBotonOrdenes		

Tabla 61 - Procedimiento punteroSobreBoton

ID	receive		
Parámetros	**Nombre**	**E/S**	**Función**
	mensaje	entrada	Contenido del paquete UDP recibido
Descripción	Este procedimiento es el handler de la librería UDP sobre eventos de recepción de datagramas. Se ha ampliado para añadir la comprobación del contenido del paquete UDP recibido, discriminando entre lecturas reenviadas y la orden de recarga de configuración.		
Procedimientos asociados	draw		

Tabla 62 - Procedimiento receive

3.3.9 PROCEDIMIENTOS DE LA PÁGINA WEB

ID	**menu.php**		
Parámetros	**Nombre**	**E/S**	**Función**
	condiciones	entrada	Archivo de configuración de condiciones *condiciones.txt*
Descripción	Muestra el menú de contenidos de la página web. Si se detecta un estado de alerta al analizar el archivo de condiciones actuales del invernadero, se añade un icono informativo al botón *Estado Actual* para informar al usuario de que se requiere su atención.		
Elementos asociados	ultimaslect.php estadisticas.php actual.php consumoagua.php temperaturas.php index.php estilo.css		

Tabla 63 - menu.php

ID	**estadisticas.php**		
Parámetros	**Nombre**	**E/S**	**Función**
	Luminosidad	entrada	Casilla de selección de luminosidad
	Nivel de agua	entrada	Casilla de selección de nivel de agua
	Humedad del suelo	entrada	Casilla de selección de humedad del suelo
	Humedad ambiental	entrada	Casilla de selección de humedad ambiental
	Temperatura exterior	entrada	Casilla de selección de temperatura exterior
	Temperatura interior	entrada	Casilla de selección de temperatura interior
	Riego activo	entrada	Casilla de selección de actividad de riego
	Ventilación	entrada	Casilla de selección de actividad de ventilación
	valor	entrada	Selector del número de registros a utilizar para dibujar el gráfico. El valor puede ser una de las siguientes opciones mostradas al usuario en la lista desplegable: • 72 (medio día) • 144 (un día) • 288 (dos días) • 586 (cuatro días)
Descripción	Muestra en pantalla la sección *Estadísticas* usando la selección de datos suministrados por el usuario a través de la lista desplegable y las casillas de selección individual.		
Elementos asociados	menu.php sketchEstadisticas.pde		

Tabla 64 - estadisticas.php

ID	ultimaslect.php		
Parámetros	**Nombre**	**E/S**	**Función**
	valor	entrada	Número de registros a mostrar en cada gráfico. El usuario puede escoger un valor de los siguientes en la lista desplegable: • Última hora • Últimas dos horas • Últimas cuatro horas • Últimas ocho horas • Últimas doce horas
Descripción	Muestra en pantalla la sección *Ultimas lecturas* a partir de los datos suministrados por el usuario que son recogidos de la lista desplegable y las casillas de selección disponibles.		
Elementos asociados	menu.php sketchConsulta.pde		

Tabla 65 - ultimaslect.php

ID	actual.php
Parámetros	-
Descripción	Muestra en pantalla la sección *Estado actual* en la que el usuario puede visualizar la lectura mas reciente enviada desde Arduino y el estado actual de las condiciones y sistemas del invernadero. Esta página se autorefresca cada 30 segundos sin intervención del usuario.
Elementos asociados	menu.php ultimosdatos.php estadoactual.php

Tabla 66 - actual.php

ID	temperaturas.php		
Parámetros	**Nombre**	**E/S**	**Función**
	intervalo	entrada	Número de días sobre los que calcular estadísticas de temperatura
Descripción	Muestra en pantalla la sección *Temperaturas* en la que el usuario puede acceder a diversa información relacionada con las temperaturas mínimas, máximas y medias (tanto exteriores como interiores al invernadero) calculadas sobre un número de días ajustable por el usuario.		
Elementos asociados	menu.php sketchConsulta.pde		

Tabla 67 - temperaturas.php

ID	consumoagua.php
Parámetros	-
Descripción	Muestra en pantalla la sección *Consumo de agua* en la que aparece información relativa al nivel de agua en el depósito, consumo total, consumo medio diario y previsión de duración de agua restante.
Elementos asociados	menu.php sketchDeposito.pde

Tabla 68 - consumoagua.php

ID	index.php
Parámetros	-
Descripción	Muestra en pantalla la sección *Sobre Hortduino* en la que se muestra al usuario un breve resumen informativo de los objetivos del sistema Hortduino, las herramientas disponibles y el modo de funcionamiento interno del sistema.
Elementos asociados	menu.php

Tabla 69 - index.php

ID	estadoactual.php		
Parámetros	**Nombre**	**E/S**	**Función**
	condiciones.txt	entrada	Estado actual de las condiciones registradas en el invernadero y actividad de los sistemas de riego y ventilación
	config.txt	entrada	Archivo de configuración de rangos aceptables para variables y modos de control de los sistemas de riego y ventilación
Descripción	Página auxiliar contenida dentro de la sección *Estado actual* en la que se muestra el estado de las condiciones del invernadero y la configuración y actividad de los sistemas de riego y ventilación mediante imágenes representativas. El usuario puede mover el cursor del ratón encima de los elementos gráficos para obtener una explicación del significado de la imagen mostrada.		
Elementos asociados	actual.php		

Tabla 70 - estadoactual.php

ID	ultimosdatos.php
Parámetros	-
Descripción	Página auxiliar contenida dentro de la sección *Estado actual* que se muestra la última lectura recibida de los sensores, almacenada en la tabla *actual* de la base de datos del servidor.
Elementos asociados	actual.php

Tabla 71 - ultimosdatos.php

ID	sketchDeposito.pde		
Parámetros	**Nombre**	**E/S**	**Función**
	valor	entrada	Nivel actual del depósito de agua para riego
Descripción	Dibuja en el elemento *canvas* de la página web la representación gráfica del nivel de agua presente en el depósito.		
Elementos asociados	consumoagua.php		

Tabla 72 - sketchDeposito.pde

ID	sketchConsulta.pde		
Parámetros	**Nombre**	**E/S**	**Función**
	id	entrada	Nombre de la variable a graficar
	num	entrada	Número de registros a utilizar en el gráfico
Descripción	Dibuja en el elemento *canvas* de la página web la representación gráfica del la variable seleccionada, empleando para ello el número de registros indicado. También muestra los valores que componen el gráfico si el usuario mueve el puntero del ratón sobre el área del gráfico.		
Elementos asociados	ultimaslect.php		

Tabla 73 - sketchConsulta.pde

ID	sketchEstadisticas.pde		
Parámetros	**Nombre**	**E/S**	**Función**
	seleccionados	entrada	Conjunto de variables seleccionadas para mostrar en el gráfico
	num	entrada	Número de registros a utilizar en el gráfico
Descripción	Este sketch se ocupa de las labores de representación del conjunto de variables escogido por el usuario dentro del elemento *canvas* de la página *Estadisticas.php*. Las tareas que realiza el sketch son: • Obtener las marcas de tiempo de los registros almacenados en la base de datos en función del intervalo elegido. • Dibujar los ejes, rejilla y las escalas asociadas en función de las variables a mostrar en el gráfico. • Trazar las representaciones gráficas del conjunto de variables suministradas empleando los colores asociados a cada variable.		
Elementos asociados	estadisticas.php		

Tabla 74 - sketchEstadisticas.pde

ID	estilo.css
Parámetros	-
Descripción	Archivo de hoja de estilos en cascada (CSS) utilizado para controlar la apariencia global de la página web mediante reglas de estilo aplicadas a cada elemento estructural.
Elementos asociados	menu.php index.php estadisticas.php ultimaslect.php consumoagua.php temperaturas.php actual.php estadoactual.php ultimosdatos.php

Tabla 75 - estilo.css

3.4 DISEÑO DE LAS INTERFACES DE USUARIO

En esta sección se analiza la fase de diseño de las interfaces de usuario presentes en el sistema Hortduino desde el concepto inicial hasta su implementación final. Cada herramienta disponible posee sus características distintivas que influyen en el diseño de la interfaz de usuario, ya sea por el método de interacción con la aplicación (entrada táctil o mediante ratón y teclado) o por la forma de presentación de la información en pantalla.

3.4.1 DISPOSITIVO ARDUINO

El dispositivo Arduino no posee como tal una interfaz de usuario. La única forma de comunicación disponible con el usuario es mediante el empleo de programas o herramientas que permitan visualizar los mensajes de depuración emitidos por Arduino, utilizados en tareas de corrección de errores y pruebas del código que ejecuta en el microcontrolador.

Para acceder a esta interfaz, se necesita establecer una conexión entre el puerto serial de Arduino mediante el cable USB apropiado y la herramienta encargada de recibir los mensajes vía serial (por ejemplo, *PuTTY*).

3.4.2 APLICACIÓN DE ESCRITORIO

La aplicación de escritorio debe ser capaz de ofrecer en su interfaz la información de las condiciones recientes y actuales del invernadero, así como proporcionar elementos para interactuar con las condiciones aceptables de las variables y los métodos de control de los sistemas del invernadero.

Con estos dos objetivos en mente, la interfaz ha de integrar los siguientes elementos visuales:

- Información de los datos mas actuales sobre las variables monitorizadas
- Información de la evolución reciente de los datos sobre cada variable
- Información y control del modo de funcionamiento de los sistemas de riego y ventilación, así como su actividad en ese momento.
- Información sobre la disponibilidad de agua para riego en el depósito
- Información sobre el estado de las condiciones del invernadero, alertas incluidas.
- Información y control de los rangos establecidos para cada variable.
- Información y control sobre el intervalo de actualización de la información reciente.

Mediante el empleo del lenguaje *Processing* y aprovechando las funcionalidades que ofrece en lo relacionado con la creación de entornos visuales, se realizó una primera aproximación al diseño de la interfaz de la aplicación de escritorio, comenzando por mostrar la información de los datos recibidos desde Arduino en la ventana de la aplicación de escritorio.

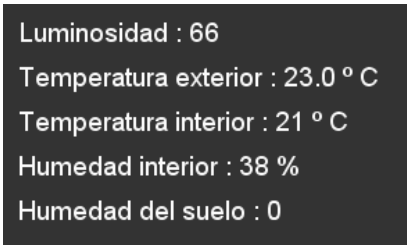


Ilustración 21 - Información actual de los sensores

A medida que se incorporaron los nuevos elementos visuales, se hizo patente la necesidad de adaptar la información mostrada a la manera mas visual posible, optando por el empleo de elementos gráficos cuando se considerase apropiado, con objeto de crear una interfaz intuitiva y de fácil comprensión por parte del usuario.

Siguiendo esta idea, los datos referidos a la evolución reciente de las variables supervisadas se adaptaron a un diseño en forma de gráfico continuo,permitiendo apreciar visualmente el comportamiento y los cambios en el valor de las variables mediante la interacción del cursor del ratón con el área de los gráficos. Se optó por la agrupación de las variables monitorizadas en dos gráficos, en base a criterios de relación, por dos motivos: evitar la sobrecarga de la ventana de información y facilitar la comparación de variables que comparten algún tipo de relación, como la unidad de medida común.

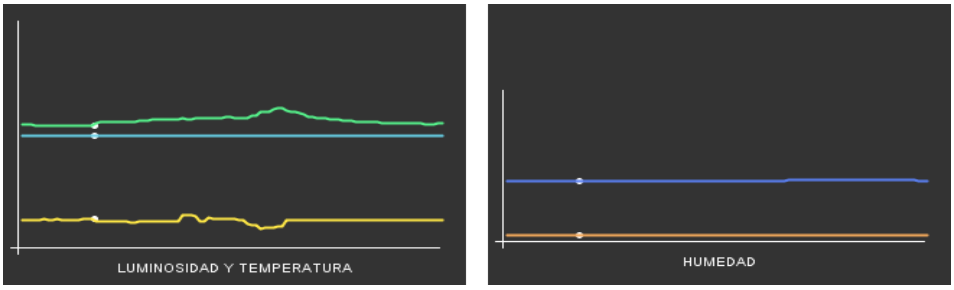


Ilustración 22 - Gráficos de evolución temporal de las variables

Por ese motivo, en el gráfico evolutivo de las temperaturas se incluye la luminosidad presenten en el invernadero, ya que la radiación solar tiene una relación directa con la temperatura del interior y exterior del invernadero. La humedad del suelo se presenta junto con la humedad ambiental en el mismo gráfico para observar el efecto de la transpiración de las plantas y la evaporación del agua presente en el suelo hacia la atmósfera interior del invernadero.

Para incorporar a la interfaz los elementos de control e información de la actividad de los sistemas remotos de riego y ventilación, se optó por un conjunto de botones acompañados de animaciones que representasen su funcionamiento o no en ese instante.

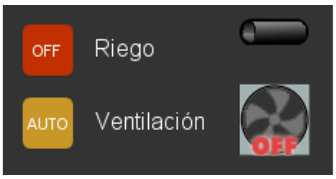


Ilustración 23 - Controles y actividad de los sistemas remotos

Cada botón refleja con el color y el texto interior el modo de control del sistema correspondiente, pudiendo establecerse en manual (ON,OFF) o automático (AUTO). Si el sistema se encuentra en funcionamiento, la imagen del lado derecho mostrará dicha actividad en forma de animación como gotas que caen del tubo o por el giro del ventilador.

Con el objetivo de informar al usuario del nivel de agua disponible para riego de forma visual, se diseñó una representación sencilla del depósito junto al nivel actual de agua contenida en el mismo. El nivel representado varía su altura dependiendo del valor porcentual que se reciba desde el sensor y el porcentaje numérico es mostrado justo por encima del nivel porcentual actual. El color del relleno del depósito varía su transparencia basado en el nivel de contenido actual (mayor transparencia a menor nivel) para mejorar la representación, contribuyendo a la percepción del nivel por el usuario.

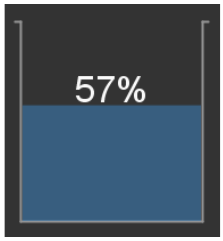


Ilustración 24 - Nivel de agua en depósito

En el caso de las condiciones del invernadero, se aplica la misma filosofía visual que hemos visto hasta ahora. La representación del estado de las condiciones actuales del invernadero emplea un conjunto de líneas de texto que muestran el estado actual junto a las imágenes descriptivas asociadas a ese estado. Variando el color de la línea informativa y la imagen mostrada, se alerta al usuario si alguna condición establecida no se cumple por no llegar al valor mínimo (color de línea azul) o superar el límite máximo permitido (color rojo)

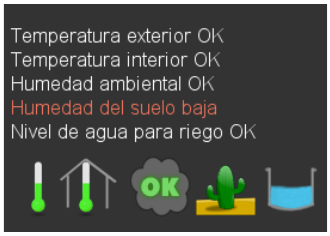


Ilustración 25 - Información sobre condiciones del invernadero

Como último elemento de la interfaz, se presentan los diferentes selectores que indican los valores máximo y mínimo permitidos para cada variable monitorizada, así como el control del intervalo de actualización de los gráficos vistos anteriormente.

El usuario puede cambiar el valor mostrado en cada caja mediante los triángulos dispuestos en la parte derecha del valor mostrado. Se puede mantener el botón del ratón pulsado sobre el triángulo para incrementar o decrementar el valor de forma continuada.

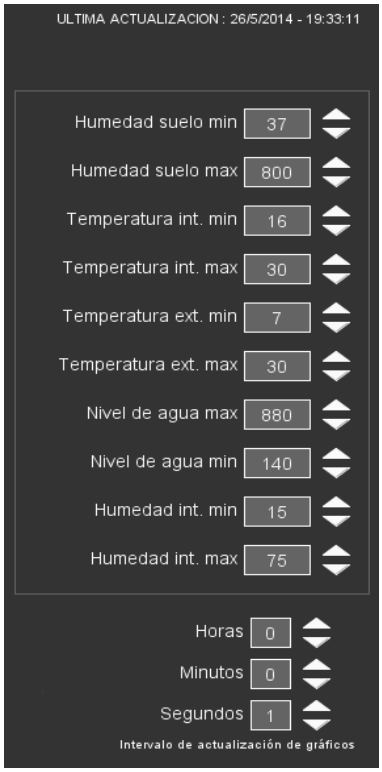


Ilustración 26 - Selectores

Si la conexión al dispositivo Arduino no fuese posible, la interfaz mostraría un mensaje de información como el de la Ilustración 25 con los datos de conexión usados y ofrecería la posibilidad de ajustarlos haciendo click en el botón *Configurar Conexión*.

El aspecto en conjunto de la interfaz de la aplicación de escritorio se puede apreciar en la Ilustración 26 de la página siguiente. Con todos los elementos comentados anteriormente, se ha querido obtener una interfaz sencilla pero completa, intuitiva y de fácil manejo para el usuario de la aplicación.

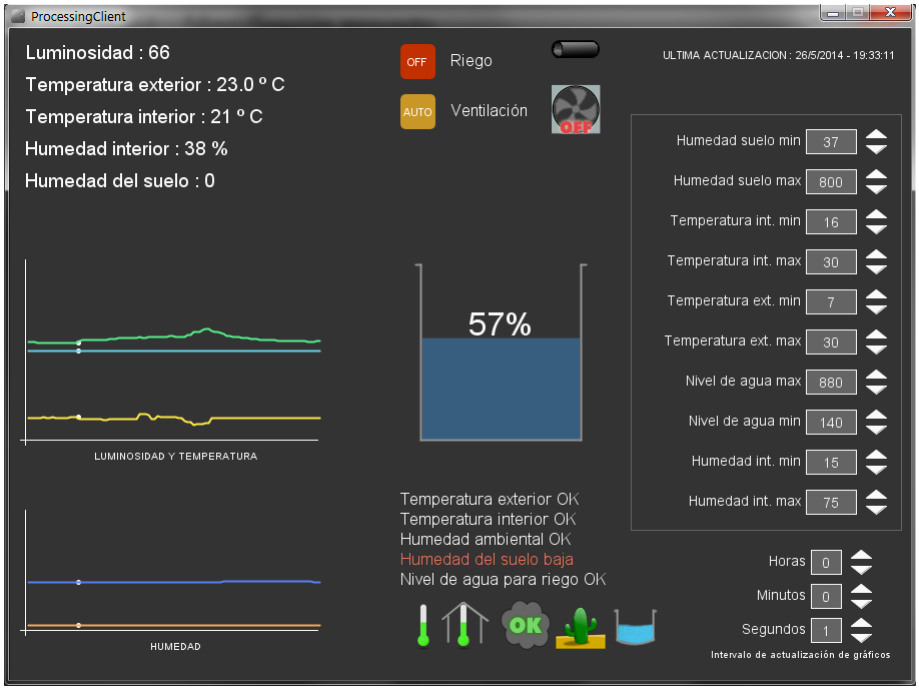


Ilustración 27 - Interfaz de la aplicación de escritorio

Ilustración 28 - Error de conexión

3.4.3 APLICACIÓN ANDROID

La interfaz de la aplicación para dispositivos Android se diseñó con el objetivo de asemejarse a la de la aplicación de escritorio, dado que cumplen casi las mismas funciones, pero adaptada a la entrada táctil. De esta manera, el usuario familiarizado con la aplicación de escritorio se adaptará rápidamente al uso de la aplicación Android a través de una interfaz con diseño y elementos muy similares.

Existen diferencias en la disposición de los elementos en la interfaz de la aplicación Android respecto a la de la aplicación de escritorio, ya que el tamaño de la pantalla de estos dispositivos es notablemente inferior al de un monitor convencional para PC. Por ello, se ha optado por incluir algunos de los elementos de la interfaz dentro de un menú sencillo para una correcta visualización de la información.

Ilustración 29 - Pantalla inicial de la aplicación Android

Como se puede observar en la ilustración anterior, la interfaz de la aplicación Android comparte numerosos elementos de diseño e información con la aplicación de escritorio. La interacción del usuario con los botones de control de riego y ventilación se realiza por vía táctil. El elemento extra de la interfaz es el botón Menu, situado en la parte inferior de la pantalla, que da acceso a otras opciones informativas y de control, como se verá a continuación.

El menú de la aplicación Android ofrece al usuario la posibilidad de visualizar en pantalla el nivel de agua en el depósito de riego, modificar los rangos aceptables de las condiciones del invernadero y acceder a información sobre la conexión con la aplicación de escritorio.

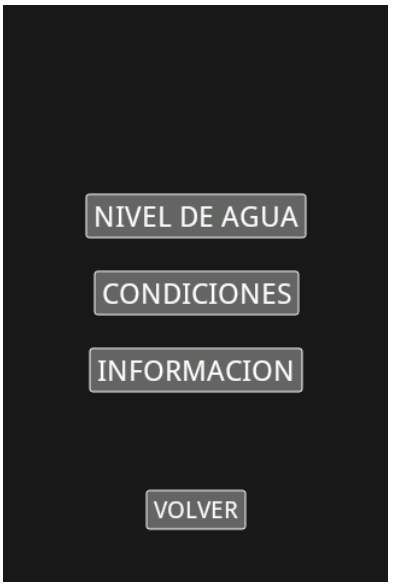


Ilustración 30 - Menú de la aplicación Android

Al igual que en la aplicación de escritorio, la representación del nivel actual de agua en el depósito de riego se muestra con el mismo esquema visual. En las siguientes ilustraciones se puede apreciar el efecto de transparencia sobre el color del relleno del depósito con niveles diferentes

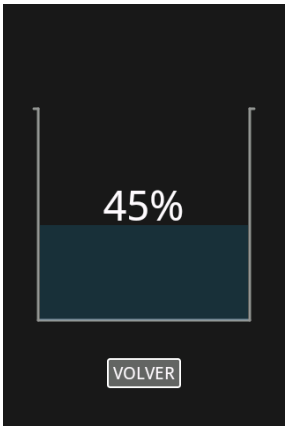


Ilustración 31 - Nivel de agua en la aplicación Android (45%)

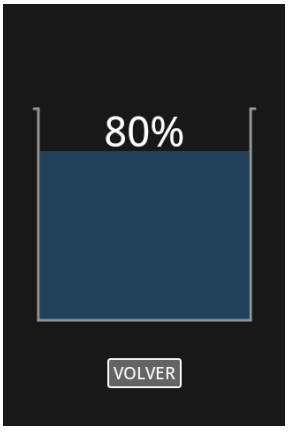


Ilustración 32 - Nivel de agua en la aplicación Android (80%)

Al acceder a la opción del menú *Condiciones*, se muestran al usuario los selectores de rangos aceptables de las variables. Este elemento ha sido rediseñado con respecto a su homólogo en la aplicación de escritorio para adaptarlo al tamaño y resolución de la pantalla del dispositivo Android. La forma en la que se alteran los valores de los rangos también ha sido rediseñada: ahora el usuario desliza el triángulo indicador de máximo o mínimo sobre una barra correspondiente a la condición a configurar. Mientras se desliza el triángulo, el valor correspondiente al máximo o mínimo va cambiando según la posición del indicador.

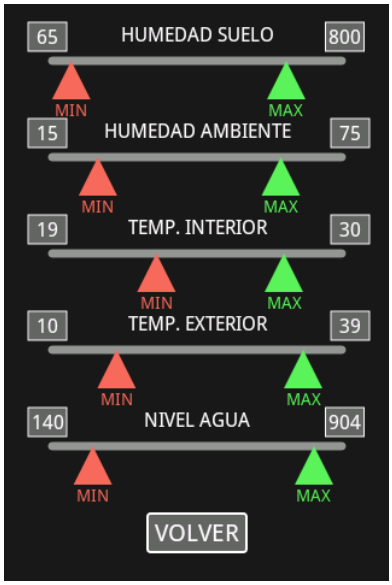


Ilustración 33 - Configuración de rangos en la aplicación Android

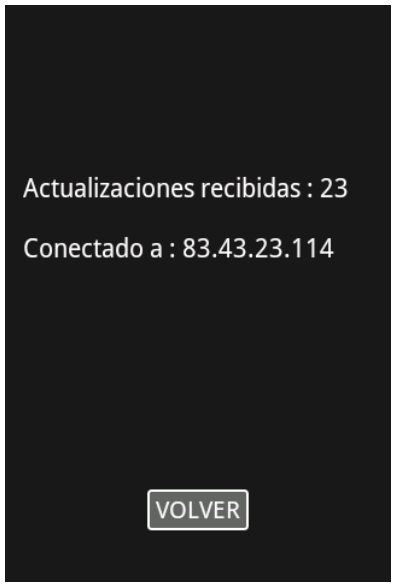


Ilustración 34 - Información de conexión con ap. escritorio

La opción del menú bajo el título *Información* muestra los datos de conexión con la aplicación de escritorio mediante dos líneas de texto: la primera contiene la dirección IP desde la que se reciben las lecturas de los sensores y la segunda muestra el número de lecturas recibidas.

En caso de que la conexión con la aplicación de escritorio no se pudiese establecer o se viese interrumpida, se mostraría una pantalla de información como la que aparece en la ilustración de la derecha, indicando la dirección IP y puerto desde los que se espera recibir las lecturas de datos.

Ilustración 35 - Pantalla de problemas de conexión

3.4.4 PÁGINA WEB

Por sus características, la página web constituye un caso especial en lo relativo a las interfaces del proyecto Hortduino, ya que no cuenta con elementos capaces de cambiar la configuración o el modo de funcionamiento de los sistemas remotos. La interacción con el usuario se limita únicamente a visualizar la información disponible en el sistema, por lo que se convierte en un medio de acceso a los datos históricos del invernadero.

El en diseño de los elementos informativos de la página web se ha empleado el lenguaje *Processing*, usado en la implementación de las aplicaciones de escritorio y Android, adaptado a las tecnologías web. Esta variante, conocida como *Processing.js* es similar en funciones a *Processing* pero tiene ciertas restricciones derivadas del uso del elemento *canvas* de HTML5, lo que limita las funciones nativas utilizables de *Processing.*

Para continuar con el estilo de las interfaces vistas en la aplicación de escritorio y Android, los *sketches* utilizan una combinación de colores e imágenes informativas similar a la de los elementos presentes en dichas interfaces.

La estructura de la página web se basa en un diseño a dos columnas donde la columna izquierda contiene el menú de opciones y la columna derecha muestra el contenido generado dinámicamente para dicha opción.

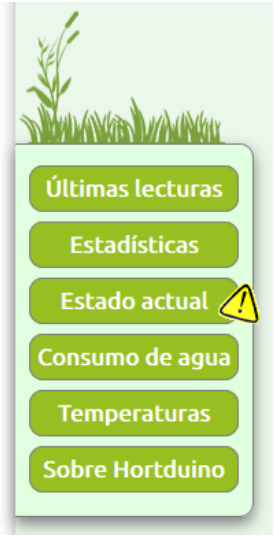


Ilustración 36 - Menú de la página web

En la ilustración de la izquierda se muestra la estructura del menú de opciones, según aparece en la página. Cada sección viene representada por un botón que enlaza a la página correspondiente.

Para informar al usuario de las alertas activas en el sistema, la sección *Estado actual* del menú presenta un icono parpadeante en caso de que se detecte una condición anómala. Esto puede ser de utilidad para informar inmediatamente al usuario cuando accede a la página web de que se necesita su atención urgentemente para corregir el problema.

La sección mostrada al ingresar en la página es *Sobre Hortduino*, donde se explica brevemente el objetivo del sistema Hortduino, sus componentes y modo de funcionamiento.

A continuación se detalla cada sección del menú de opciones, explicando los aspectos de diseño reseñables.

Últimas lecturas

Esta sección tiene como función ofrecer al usuario una visión del comportamiento en el tiempo de las condiciones del invernadero en las últimas 12 horas. Para ello se han diseñado una serie de *sketches* que generan un gráfico con los valores presentados por la variable en cuestión a partir del intervalo a mostrar y la información almacenada en la base de datos.

El usuario puede escoger el espacio de tiempo a representar en el gráfico y tiene la posibilidad de mover el cursor del ratón sobre el gráfico para obtener el valor particular sobre ese punto del gráfico. En la ilustración 34 se muestra el contenido y la estructura de la sección *Ultimas lecturas* como se vería en el navegador del usuario.

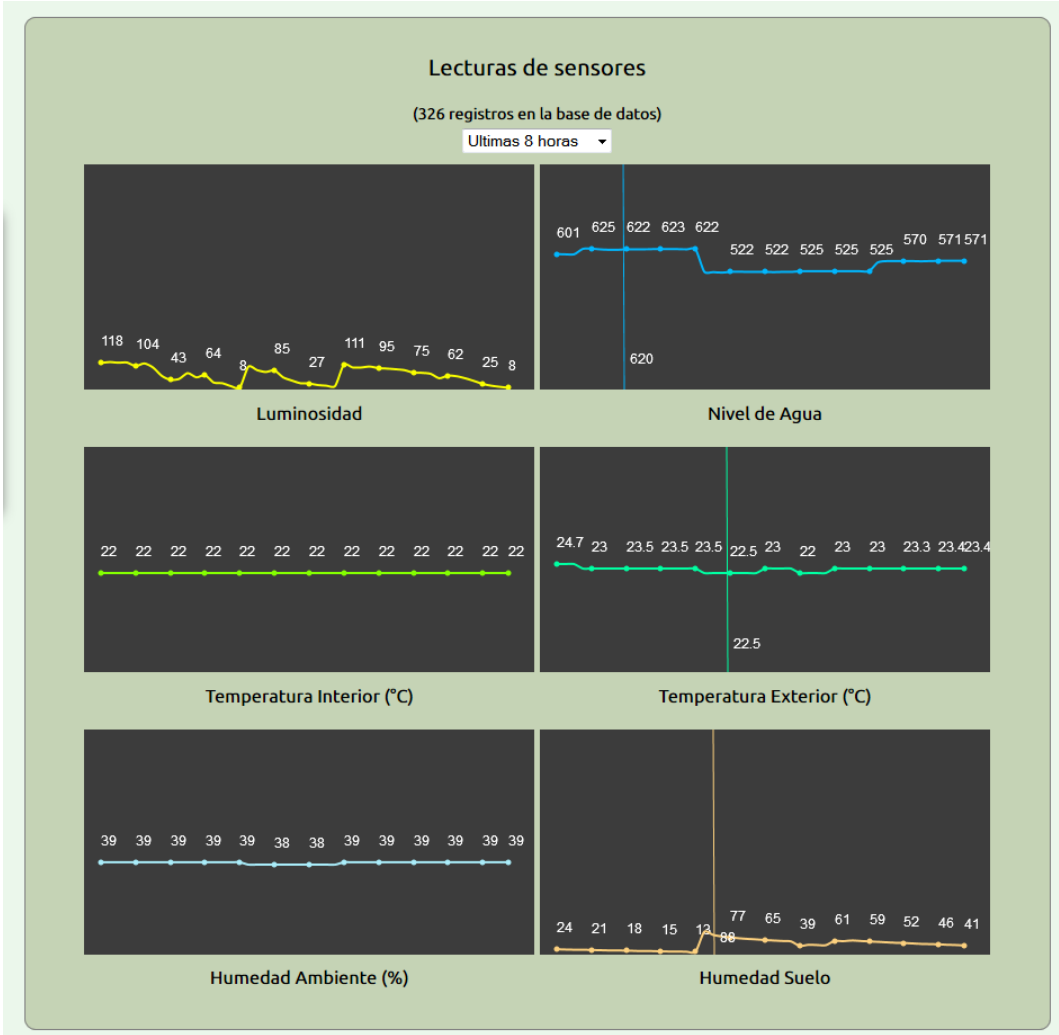


Ilustración 37 - Sección *Ultimas Lecturas* de la página web

Estadísticas

El diseño de esta sección intenta responder a la necesidad de representar la evolución de varias variables en un mismo gráfico, lo que permite al usuario analizar el comportamiento de unas variables frente a otras. A modo de ejemplo, el usuario puede mostrar la evolución de la luminosidad frente a las temperaturas interior y exterior registradas en el invernadero, obteniendo así información acerca de la capacidad de aislamiento térmico que proporciona el invernadero frente a las condiciones exteriores cuando no hay luz solar disponible.

Esta funcionalidad se ve ampliada frente a la ofrecida en la sección *Últimas lecturas* al disponer de un rango temporal mayor en el que mostrar la evolución de las variables supervisadas, así como la posibilidad de mostrar los periodos de actividad de los sistemas de riego y ventilación.

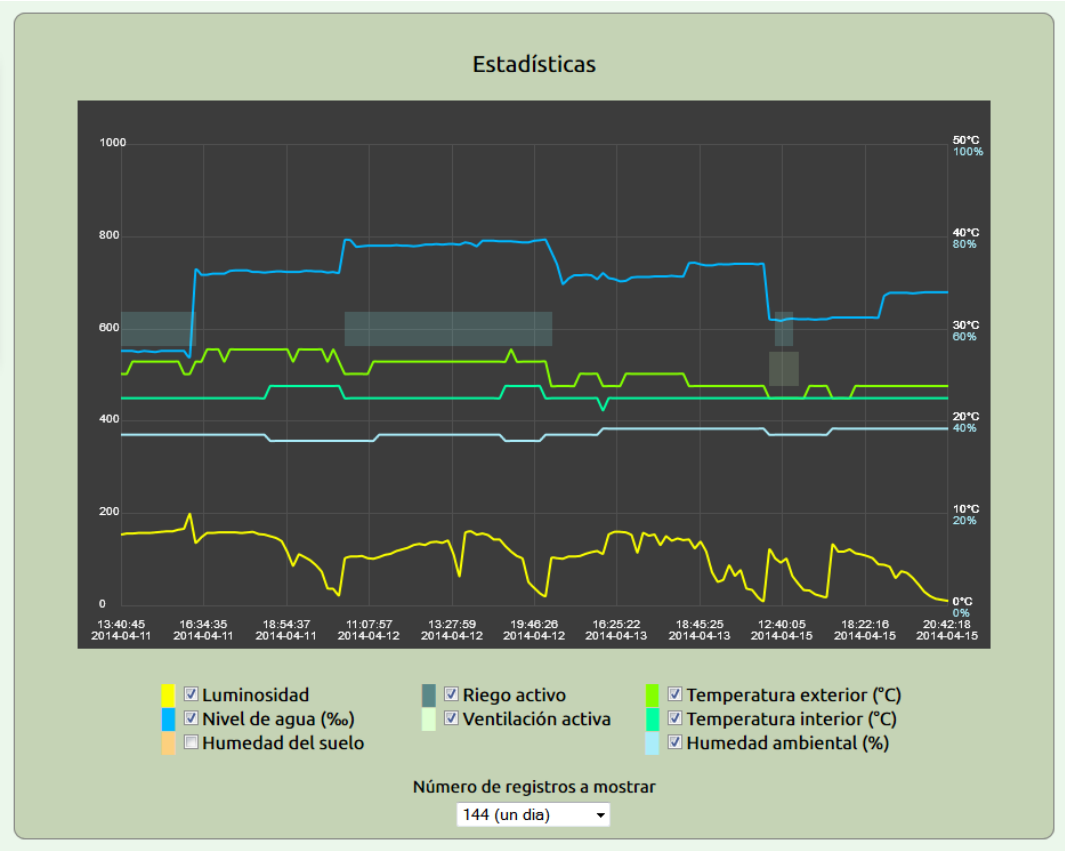


Ilustración 38 - Sección *Estadísticas* de la página web

Estado actual

Esta sección se ha diseñado para incorporar a la web la información mas reciente recibida por la aplicación escritorio, así como el estado actual de las condiciones y sistemas del invernadero. Se ha intentado mantener la apariencia establecida en las interfaces de las aplicaciones de escritorio y Android para crear una sensación de uniformidad en el sistema de cara al usuario. Los *sketches* dispuestos en este apartado replican la información ofrecida por los elementos de la interfaz de la aplicación de escritorio, pero adaptados a la visualización en la web. No disponen de la capacidad de interactuar con las condiciones del sistema.

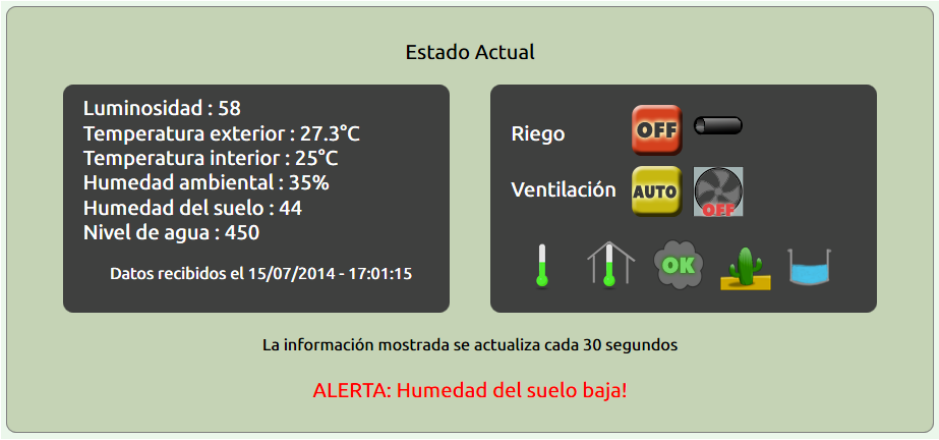


Ilustración 39 - Sección *Estado Actual* de la web

Consumo de agua

La idea detrás de esta sección es mostrar al usuario información útil acerca del nivel de agua para riego disponible en el depósito. Al valor porcentual del nivel, mostrado en las interfaces de la aplicación de escritorio y Android, se suma el consumo total de agua registrado, el consumo medio por día y la duración estimada del agua restante en el depósito.

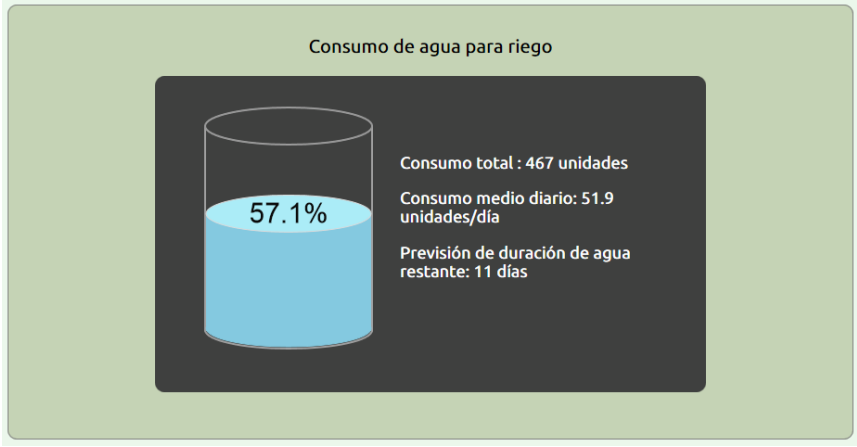


Ilustración 40 - Sección *Consumo de Agua* de la página web

Temperaturas

Esta sección está pensada para mostrar al usuario información acerca de los valores significativos de las temperatura registradas por los sensores. Se ofrece información sobre la temperatura interior y exterior mínima, máxima y media en el intervalo de tiempo que introduzca el usuario en el campo dispuesto a tal fin.

Ilustración 41 - Sección *Temperaturas Registradas* de la página web

Sobre Hortduino

Aquí se explica brevemente el objetivo del sistema Hortduino, su funcionamiento y las diferentes partes que lo componen. Esta sección es la primera en visualizarse por el usuario cuando se accede a la web en el navegador y es la única no generada dinámicamente por el servidor.

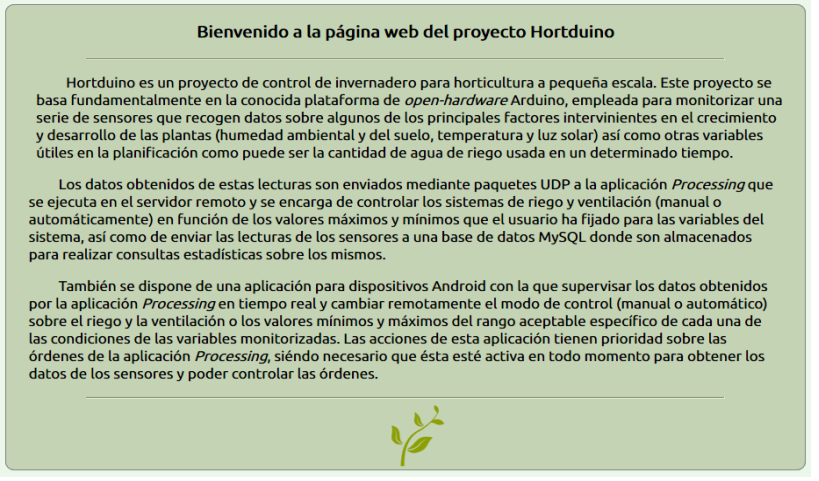


Ilustración 42 - Sección *Sobre Hortduino* de la página web

3.5 ENTORNO TECNOLÓGICO DEL SISTEMA

El entorno tecnológico en el que se basa el sistema Hortduino se puede dividir en tres partes diferenciadas, asociadas a cada componente específico del sistema:

- **Dispositivo Arduino MEGA 2560**

El hardware Arduino se basa en un conjunto de componentes electrónicos ensamblados sobre una placa y conectados a un microcontrolador, el cual se puede programar para que lleve a cabo operaciones mas o menos complejas e interactúe con elementos externos conectados a este hardware.

Para la tarea de programación del microcontrolador, Arduino cuenta con un entorno de desarrollo integrado propio que utiliza el lenguaje *C/C++* y el compilador *avr-gcc*, proporcionando capacidades de depuración y carga de programas a la memoria *flash* del microcontrolador. Debido a la limitación de memoria disponible asociada al uso de un microcontrolador, los programas a ejecutar no deben sobrepasar los 248 KB de tamaño total y se ven restringidos a usar una cantidad máxima de memoria de 8 KB para operar con datos en tiempo de ejecución.

La comunicación de la aplicación de escritorio con Arduino se efectúa mediante el empleo de un *shield Ethernet* acoplado al dispositivo Arduino, el cual proporciona las funciones básicas de conectividad en red local. La interfaz serial de Arduino se utiliza para labores de depuración de código o pruebas.

- **Aplicaciones de escritorio y Android**

La aplicación de escritorio y la aplicación para dispositivos Android se han implementado usando el lenguaje *Processing*, derivado de *Java,* con sintaxis simplificada y modelo de programación visual. Los programas en *Processing* se denominan *sketches* y son subclases de la clase *PApplet* de *Java* que implementa las interfaces relacionadas con eventos de ratón y teclado. Para la ejecución de los programas, *Processing* hace uso de la máquina virtual de *Java*, por lo que ofrece las características asociadas a ella como la compatibilidad con los sistemas operativos mas usados y la independencia del procesador presente en el dispositivo que ejecute el programa.

- **Página Web y Servidor XAMPP**

La página web del proyecto Hortduino hace uso de los elementos disponibles en el lenguaje HTML5 combinados con *sketches* implementados en *Processing.js* para generar dinámicamente contenido informativo que sea de utilidad al usuario. Las herramientas necesarias para que esto pueda llevarse a cabo son provistas por el servidor XAMPP en forma de servicio de motor de base de datos (MySQL) y servidor web (Apache).

4 IMPLEMENTACIÓN

En esta sección se muestran los fragmentos de código y comentarios mas destacables, relativos a las diferentes aplicaciones que componen el sistema Hortduino, para ampliar lo mostrado en el apartado 7 *Cuestiones de diseño e implementación reseñables* del documento correspondiente al Bloque I de esta memoria.

CLASE BOTÓN DE LA APLICACIÓN ANDROID

Esta sencilla clase implementa el objeto *Botón* usado en la interfaz de la aplicación para dispositivos Android. También establece los métodos para dibujar el botón y controlar cuándo se ha presionado.

```
class Boton {

  float xPos;
  float yPos;
  int ancho;
  int alto;
  int tamFuente;
  String texto;
  int margen;

  Boton (float _xPos, float _yPos, String _texto, int _tamFuente, int _margen) {

    xPos = _xPos;
    yPos = _yPos;
    texto = _texto;
    tamFuente = _tamFuente;
    margen = _margen;
    textSize (tamFuente);
    alto = tamFuente + margen*2;
    ancho = (int)textWidth (texto) + margen*2;
    textSize (tamText);
  }
  //--------------------------------------------------------------------------

  boolean esPresionado () {

    float x, y;

    x = xPos - ancho/2;          // El punto (x,y) referencia la esquina superior
                                    izquierda del botón
    y = yPos - alto/2;

    if ((presionado) && (mouseX >= x) && (mouseX <= (x + ancho)) && (mouseY >= y)
&& (mouseY <= (y + alto)))
      return true;
    else
      return false;
  }

  //--------------------------------------------------------------------------
```

```
  void dibujarBoton () {        // La referencia para dibujar el botón es su centro

    float x, y;

    x = xPos - ancho/2;         // El punto (x,y) referencia la esquina superior
                                   izquierda del botón
    y = yPos - alto/2;

    stroke (255);
    fill (100);
    rect (x, y, ancho, alto, margen/2);
    fill (250);
    textSize(tamFuente);
    text (texto, xPos - textWidth(texto)/2, yPos + tamFuente/2 - margen/2);
    textSize (tamText);
  }
}
```

CLASE SELECTOR DE LA APLICACIÓN ANDROID

La clase *Selector* es la encargada de proporcionar las funcionalidades necesarias para dibujar en pantalla el selector asociado a cada variable y modificar los valores máximo y mínimo a voluntad del usuario, empleando para ello la entrada táctil del dispositivo Android.

```
class Selector {

  float xPos;
  float yPos;
  int ancho;
  int alto;
  int minimo;
  int maximo;
  int margen;
  int limiteMin;
  int limiteMax;
  int id;
  int tamFuente;
  String texto;

  Selector (float _xPos, float _yPos, String _texto, int _tamFuente, int _margen,
              int _id, int _limiteMin, int _limiteMax, int _minimo, int _maximo)
 {
    xPos = _xPos;
    yPos = _yPos;
    texto = _texto;
    margen = _margen;
    minimo = _minimo;
    maximo = _maximo;
    limiteMin = _limiteMin;
    limiteMax = _limiteMax;
    tamFuente = _tamFuente;
    id = _id;
    textSize (tamFuente);
    ancho = displayWidth - (2 * margen);
    alto = 2 * tamFuente;
  }

  //--------------------------------------------------------------------------------
```

```
  private String selectorPresionado (float x, float y, int ancho, int idnt, String
maxmin) {
// Devuelve en un string el ID del selector presionado

    String resultado = "NONE,0";

    if (punteroSobreBoton (int(x-ancho/2), int(y), ancho, ancho) && (idnt ==
       this.id)) {
      resultado = maxmin + "," + idnt;
      sltrPresionado = true;            // Indica que hay un selector que está
                                           siendo presionado
    }
    return resultado;
  }

  //--------------------------------------------------------------------------

  private void dibujarMarcador (float x, float y, int ancho, String identmaxmin,
               int idSelector) {
 // Muestra los triángulos indicadores sobre la línea

    if (!sltrPresionado) selectorpres = selectorPresionado (x, y, ancho, this.id,
       identmaxmin);

    String[] cadena = split(selectorpres, ',');

    if ( (touchEvent == "MOVE") && (!cadena[0].equals("NONE")) ) {

      noStroke();

      if ( (int(cadena[1])==this.id) && (identmaxmin=="MIN") &&
        (cadena[0].equals("MIN")) ) {
               fill (240, 104, 89);
               x = constrain (mouseX, margen, displayWidth - margen);
               triangle (x, y, x-ancho/2, y+ancho, x+ancho/2, y+ancho);
                stroke (250);
               textSize (tamFuente-2);
               text (identmaxmin, x-textWidth(identmaxmin)/2, y+ancho+tamFuente-2);
        // Muestra MAX o MIN debajo del triangulo marcador
                minimo = (int) map (mouseX, margen, displayWidth-margen, limiteMin,
                           limiteMax);
               minimo = constrain (minimo, limiteMin, maximo);
              arrayCondiciones[id+1] = minimo;
      }

      if ( (int(cadena[1])==this.id) && (identmaxmin == "MAX") &&
        (cadena[0].equals("MAX")) ) {
               fill (89, 240, 89);
               x = constrain (mouseX, margen, displayWidth - margen);
               triangle (x, y, x-ancho/2, y+ancho, x+ancho/2, y+ancho);
               stroke (250);
               textSize (tamFuente-2);
               text (identmaxmin, x-textWidth(identmaxmin)/2, y+ancho+tamFuente-2);
        // Muestra MAX o MIN debajo del triangulo marcador
               maximo = (int) map (mouseX, margen, displayWidth-margen, limiteMin,
                           limiteMax);
               maximo = constrain (maximo, minimo, limiteMax);
               arrayCondiciones[id] = maximo;
      }
    }

    if (identmaxmin == "MAX") fill (89, 240, 89);
    else fill (240, 104, 89);
    noStroke();
```

```
    triangle (x, y, x-ancho/2, y+ancho, x+ancho/2, y+ancho);
    stroke (250);
    textSize (tamFuente-2);
    text (identmaxmin, x-textWidth(identmaxmin)/2, y+ancho+tamFuente-2);
  }

  //-------------------------------------------------------------------------------

  private void dibujarCajaTexto (float x, float y, int tamFuente, String valor) {

    int ancho = tamFuente*2;
    int alto = tamFuente*3/2;

    stroke (250);
    fill (100);
    strokeWeight (1);
    rect (x, y, ancho, alto);
    fill (250);
    text (valor, x + ancho/2 - textWidth(valor)/2, y + alto - tamFuente*3/8 );
  }

  //-------------------------------------------------------------------------------

  private void dibujarLinea (float x, float y) {

    stroke (150);
    strokeWeight (7);
    line (x+margen, y+tamFuente*3, displayWidth-margen, y+tamFuente*3);
    strokeWeight (2);
  }

  //-------------------------------------------------------------------------------

  void dibujarSelector () {

    stroke (250);
    text (texto, xPos + displayWidth/2 - textWidth(texto)/2, yPos + tamFuente*2);
    dibujarCajaTexto ( xPos + margen/2, yPos + tamFuente, tamFuente, str(minimo) );
    dibujarCajaTexto ( displayWidth - margen - textWidth(str(maximo))/2, yPos +
       tamFuente, tamFuente, str(maximo) );
    dibujarLinea (xPos, yPos);
    dibujarMarcador ( map(minimo, limiteMin, limiteMax, margen, displayWidth-
       margen), yPos + tamFuente*3, tamFuente*2, "MIN", id);
    dibujarMarcador ( map(maximo, limiteMin, limiteMax, margen, displayWidth-
       margen), yPos + tamFuente*3, tamFuente*2, "MAX", id);
    textSize (tamText);
    stroke (250);
    fill (250);
  }
}
```

SKETCH ESTADÍSTICAS DE LA PÁGINA WEB

Este sketch es invocado desde la sección *Estadísticas* de la página web. Su función es la de generar gráficos representativos de las variables seleccionadas por el usuario recabando los datos pertinentes desde la base de datos.

```
float zoomX = 3;                // Separación entre puntos del eje X
float etiqueta = 10;            // Espaciado de etiquetas en el gráfico
float xPos = 15;                // Posición inicial de dibujado en el eje X
float yPos = 50;                // Posición inicial de dibujado en el eje Y
int ancho = 800;
int alto = 480;
int margen = 38;                // Margen desde el borde
String[] timeStamps;
boolean grillaDibujada = false;      // Indica si se ha dibujado la grilla de fondo
boolean humAmbSeleccionada = false;  // Indica si hay que usar la escala especial
                                        del lado derecho en el eje Y
void setup() {

        size (ancho,alto);
        background (60);
        frameRate (12);
        textSize (10);
        noFill();
        noLoop();
}

//--------------------------------------------------------------------------------

void obtenerTimeStamps (int num) {    // Obtiene las marcas de tiempo del intervalo
                                         seleccionado
        String consulta = "SELECT Fecha FROM (SELECT * FROM lecturas ORDER BY
Indice DESC LIMIT "+ num +") sub ORDER BY Indice ASC";
        String[] temp = loadStrings ("http://hortduino.no-ip.biz/consultar.php?
consulta=" + consulta);
        timeStamps = splitTokens(temp[0],",");
}

//--------------------------------------------------------------------------------

void graficar (String nombre, int num) {        // Dibuja el gráfico en el sketch

        int minSensor = 0;
        int maxSensor = 0;
        byte r,g,b;             // Valores RGB del color de la línea en el gráfico
        String consulta = "SELECT " + nombre + " FROM (SELECT * FROM lecturas ORDER
               BY Indice DESC LIMIT "+ num +") sub ORDER BY Indice ASC";
        String[] temporal = loadStrings ("http://hortduino.no-ip.biz/consultar.php?
               consulta=" + consulta);
        String[] datos = split ( temporal[0], ",");
        xPos = margen;

        strokeWeight(2);
        if ( nombre.equals("Luminosidad") ) { r=250;g=256;b=0; minSensor = 0;
               maxSensor = 1000;  }
        else if ( nombre.equals("NivelAgua") ) { r=0;g=183;b=255; minSensor = 0;
               maxSensor = 1000;  }
        else if ( nombre.equals("TempInterior") ) { r=0;g=255;b=162; minSensor = 5;
               maxSensor = 50;  }          // minSensor = 5 para ajustar mejor la
                                              representación gráfica
        else if ( nombre.equals("TempExterior") ) { r=132;g=255;b=0; minSensor = 5;
```

```
maxSensor = 50;  }
        else if ( nombre.equals("HumAmbiente") ) { r=170;g=237;b=250; minSensor =
10; maxSensor = 100; humAmbSeleccionada = true; }
        else if ( nombre.equals("HumSuelo") ) { r=252;g=208;b=126; minSensor = 0;
maxSensor = 1000;  }

        dibujarGrilla (10,5, maxSensor);
        stroke (r,g,b);
        strokeWeight(2);
        zoomX = (ancho - (2*margen))/num;
        curveTightness(0.8);
        strokeJoin(ROUND);
        beginShape();
                yPos = map(int(datos[0]),minSensor,maxSensor,alto,0) - margen;
                curveVertex(xPos, yPos);
                for (int i=0; i<datos.length-1; i++) {
                        yPos = map(int(datos[i]),minSensor,maxSensor,alto,0) -
                              margen;
                        if ( (i==datos.length-2) && (xPos>=750) ) xPos = 763;
// Ajusta el fin de la línea del gráfico al borde derecho de la grilla
                        curveVertex(xPos, yPos);
                        xPos += zoomX;
                }
                curveVertex(xPos, yPos);
        endShape();
}

//-------------------------------------------------------------------------------------

void graficarRiego (int num) {          // Muestra la línea de actividad del riego

        float finLinea = margen;
        float inicioLinea = margen;

        String consulta = "SELECT EstadoRiego FROM (SELECT * FROM lecturas ORDER BY
Indice DESC LIMIT "+ num +") sub ORDER BY Indice ASC";
        String[] temporal = loadStrings ("http://hortduino.no-ip.biz/consultar.php?
consulta=" + consulta);
        String[] datos = split ( temporal[0], ",");

        strokeCap(SQUARE);
        zoomX = (ancho - (2*margen))/num;
        stroke (138,255,255,40);
        strokeWeight(30);
        for (int i=0; i<datos.length; i++) {

                if (datos[i].equals("1")) {  // Se alarga la línea hasta
                                             encontrar un 0 (inactividad)
                        finLinea += zoomX;
                        if ( (datos[i+1].equals("0")) || (i==datos.length-2) ) {
                                line(inicioLinea,200,finLinea,200);
                                inicioLinea = finLinea;
                        }
                }
                else {
                        inicioLinea += zoomX;
                        finLinea = inicioLinea;
                }
        }
}

//-------------------------------------------------------------------------------------
```

```
void graficarVent (int num) {    // Muestra la línea de actividad de la ventilación

        float finLinea = margen;
        float inicioLinea = margen;

        String consulta = "SELECT EstadoVent FROM (SELECT * FROM lecturas ORDER BY
               Indice DESC LIMIT "+ num +") sub ORDER BY Indice ASC";
        String[] temporal = loadStrings ("http://hortduino.no-ip.biz/consultar.php?
               consulta=" + consulta);
        String[] datos = split ( temporal[0], ",");

        strokeCap(SQUARE);
        zoomX = (ancho - (2*margen))/num;
        stroke (205,252,187,40);
        strokeWeight(30);
        for (int i=0; i<datos.length; i++) {

                if (datos[i].equals("1")) {    // Se alarga la línea hasta
                                                  encontrar un 0 (inactividad)
                        finLinea += zoomX;
                        if ( (datos[i+1].equals("0")) || (i==datos.length-2) ) {
                                line(inicioLinea,235,finLinea,235);
                                inicioLinea = finLinea;
                        }
                }
                else {
                        inicioLinea += zoomX;
                        finLinea = inicioLinea;
                }
        }
}

//--------------------------------------------------------------------------------

void dibujarGrilla (int divX, int divY, int maximo) {    // Rejilla del gráfico

        float intervX = (ancho-2*margen)/divX;    // Intervalos en los que dibujar
                                                  las lineas horizontales y verticales
        float intervY = (alto-2*margen)/divY;

        String texto = "";
        String fecha = "";
        String hora = "";

        int contDivY = 0;               // Contador de las divisiones en eje Y
        int contDivX = 0;               // Contador de las divisiones en eje X
        int puntero = 0;

        stroke(80);
        strokeWeight(1);
        if ( (!grillaDibujada) || (maximo<=100) ) {

           for (float i=margen; i<ancho; i+=intervX) {    // Lineas verticales
               if (contDivX < divX+1) {
                  if (!grillaDibujada) line (i,margen,i,alto-margen);
                       puntero = int(map (i,margen,ancho-
                       margen,0,timeStamps.length));
                       texto = timeStamps[puntero];
                       hora = texto.substring(texto.indexOf (" ")+1);
                       fecha = texto.substring(0,texto.indexOf (" "));
                       if (!grillaDibujada) text (hora, i-textWidth(hora)/2, alto-
                              margen+20);
                       if (!grillaDibujada) text (fecha, i-textWidth(fecha)/2,
                              alto-margen+30);
```

```
                                contDivX++;
                            }
                    }

            for (float i=margen; i<alto; i+=intervY) {   // Lineas horizontales
                 if (contDivY < divY+1) {
                   if (!grillaDibujada) line (0+margen,i,ancho-margen,i);
                      if (maximo!=0) texto = str ( maximo - (maximo/divY)*contDivY );
                          // Escala en el eje Y
                       else texto = "";
                       if ((maximo>100) && (!grillaDibujada))
                          text (texto,margen/2,i+2);
                        else if  (!humAmbSeleccionada)
// La humedad ambiental (10%-100%) no comparte escala con los sensores de
temperatura (0-50°)
                     if (maximo!=0) text (texto+"°C", ancho-margen+5,i);
// La escala de la humedad ambiental se muestra un poco por debajo de su posición
normal
                   else text (texto, ancho-margen+5,i);
                 else {
                        fill (170,237,250);
                        text (texto+"%", ancho-margen+5,i+10);
//Escala de la humedad ambiente en azul claro para una mejor visualización
                        fill (255);
                        noFill();
                       }
                contDivY++;
              }
             }
                    humAmbSeleccionada = false;
                    grillaDibujada = true;
// Evita que se dibuje tantas veces como opciones seleccionadas marque el usuario
            }
}

//-------------------------------------------------------------------------------------

void mostrarVacio(int num) {
        obtenerTimeStamps(num);
        dibujarGrilla (10,5,0);
}

//-------------------------------------------------------------------------------------

void muestraEstadisticas (String seleccionadas, int num) {

        background(60);
        grillaDibujada = false;
        String[] temporal = split (seleccionadas,",");
        if (temporal[0]=="") mostrarVacio(num);
        obtenerTimeStamps(num);
        for (int i=0; i<temporal.length-1; i++) graficar (temporal[i],num);
}
```

PAGINA WEB DE CONSUMO DE AGUA

A continuación se muestra un ejemplo de generación de página web dinámica que utiliza los sketches implementados en *Processing.js* en conjunción con los datos obtenidos de la base de datos mediante consultas SQL invocadas en PHP para visualizar en pantalla el nivel de agua en el depósito y los datos referentes al consumo.

```
<!DOCTYPE html>
    <?php
            $conexion = mysqli_connect("hortduino.no-
ip.biz","hortelano","hortelano","inver");
            if (mysqli_connect_errno()) { echo "Fallo al conectar con MySQL: " .
mysqli_connect_error(); }

            $result = mysqli_query($conexion,"SELECT NivelAgua FROM lecturas ORDER
BY Fecha");
            $row = mysqli_fetch_array($result);
            $consumo = 0;
            $anterior = $row['NivelAgua'];
            while ($row = mysqli_fetch_array($result)) {
                    if ($row['NivelAgua']<$anterior) {
                            $consumo += $anterior - $row['NivelAgua'];
                            $anterior = $row['NivelAgua'];
                    }
                    else $anterior = $row['NivelAgua'];
            }

            $result = mysqli_query($conexion,"SELECT COUNT(DISTINCT DATE(Fecha))
FROM lecturas");
            $dias = mysqli_fetch_array($result)[0];
            $consumoMedio = round($consumo/$dias,1);
            $result = mysqli_query($conexion,"SELECT NivelAgua FROM lecturas ORDER
BY Indice DESC LIMIT 1 ");
            $nivelAguaActual = mysqli_fetch_array($result)[0];
            mysqli_close($conexion);
    ?>
    <html>
     <head>
       <title>Horduino web - Consumo de agua</title>
       <meta name="author" content="Miguel Angel Hernanz">
       <meta name="description" content="Horduino Web Sensor">
       <meta name="keywords" content="hortduino,processing,arduino,sensor">
       <meta charset="UTF-8">
       <link rel="icon" type="image/png" href="/images/favicon.png">
       <link rel="stylesheet" type="text/css" href="estilo.css">
       <script type="text/javascript" src="scripts/processing-
       1.4.1.min.js"></script>
       <script type="text/javascript">
          function mostrarDeposito() {
              var pjs = Processing.getInstanceById('depositoagua');
              var nivel = <?php echo json_encode($nivelAguaActual); ?>;
                          pjs.dibujarDeposito(nivel);
          }
        </script>
     </head>
    <body onload="mostrarDeposito()">
      <iframe name="dummie" src="dummie.php" width="0" height="0"
       style="border:none; z-index:-1; position:absolute"></iframe>
```

```
        <!-- iframe falso para forzar la carga de sketches -->
       <div id="container">
         <div style="display:block; margin: 0 auto; width:65%; height:13em;
          vertical-align: middle;">
         <div id="cabecera">
                <img id="logo" src="/images/Hortlogo.png" height="150" width="150">
                 Hortduino Web Sensor
         </div>
            </div>
            <?php include("/menu.php"); ?>
            <div id="containerSketches">
            <p>Consumo de agua para riego</p>
            <div id="riego">
              <canvas style="display: inline-block; float: left" id="depositoagua"
                  data-processing-sources =
                 "scripts/sketchDeposito/sketchDeposito.pde">
              </canvas><br><br>
               <p>Consumo total : <?php echo $consumo ?> unidades</p>
               <p>Consumo medio diario: <?php echo $consumoMedio ?>
unidades/d&iacute;a</p>
               <p>Previsi&oacute;n de duraci&oacute;n de agua restante:
              <?php echo round($nivelAguaActual/$consumoMedio) ?> d&iacute;as</p>
              </div>
            <br><br>
          </div>
          <br>
          <p class="imagencentrada">Hortduino usa tecnolog&iacute;a de<br><br><img
src="images/poweredby.png" width="290" height="32" style="text-align:center;
margin: 0 auto;"></p>
            </div>
          </body>
    </html>
```

En el código anterior se hace uso del script `processing-1.4.1.min.js` que es el encargado de convertir el código escrito en *Processing* a *JavaScript* y ejecutarlo en el elemento *canvas* asociado dentro del cuerpo de la página web.

Todas las secciones de la página web de Hortduino emplean los estilos recogidos en el archivo de hojas de estilo en cascada *estilo.css* para facilitar los cambios visuales y agilizar la escritura del código fuente. En elementos puntuales, como puede ser un párrafo específico, se opta por los estilos en línea para no sobrecargar excesivamente el archivo de estilo.

5 PRUEBAS DEL SISTEMA

Con objeto de verificar que las diferentes partes del sistema cumplen las funciones requeridas correctamente y se integran sin problemas, se diseñaron una serie de pruebas con las que testear las funcionalidades y puntos críticos. Las pruebas realizadas son:

- ✔ Pruebas estructurales, en las que se centra la atención en comprobar los flujos de ejecución dentro de la aplicación software que se producen entre los procedimientos.
- ✔ Pruebas unitarias, que tienen por objetivo testear cada módulo independientemente de su relación con el resto.
- ✔ Pruebas de integración, en las cuales se comprueba el funcionamiento de todos los módulos que componen la aplicación en su conjunto global y de una sola vez.
- ✔ Pruebas de conjunto, destinadas a valorar el correcto funcionamiento de todas las aplicaciones del sistema al mismo tiempo a través de las interacciones que se producen entre ellas.

Para la realización de las pruebas, se han empleado diferentes entornos de ejecución en función de la naturaleza de la aplicación (Arduino, Android y escritorio de Windows). Para las pruebas de la aplicación Android se utilizó un dispositivo Samsung Galaxy Mini 2 debido a que el emulador provisto en el SDK de Android presentaba un funcionamiento lento y a veces no respondía de acuerdo a lo esperado, mientras que en el smartphone sí.

En el caso del dispositivo Arduino, las pruebas iniciales se efectuaron mediante la conexión serial a PC para después proceder con el testeo de envío y recepción de paquetes UDP mediante red Ethernet.

Para las pruebas de la página web se utilizó el navegador Mozilla Firefox en su versión 29 y el navegador Internet Explorer 10. Las versiones anteriores del navegador Internet Explorer pueden no soportar determinados elementos presentes en el estándar de HTML5, por lo que se desaconseja su utilización para visualizar la página web.

A continuación se presentan el listado en forma de tablas de las pruebas realizadas al sistema, divididas por aplicación.

ARDUINO		
Prueba	**Resultado**	**Validado**
Conexión vía serial con PC	Actividad registrada en el puerto serial asignado	✔
Lectura sensor de luminosidad	Lectura recibida correctamente.	✔
Lectura sensor de temperatura interior	Lectura recibida correctamente.	✔
Lectura sensor de temperatura exterior	Lectura recibida correctamente.	✔
Lectura sensor de humedad ambiental	Lectura recibida correctamente.	✔
Lectura sensor de humedad del suelo	Lectura recibida correctamente.	✔
Lectura del sensor ultrasónico	Lectura recibida correctamente.	✔
Activación del sistema de riego	El sistema se activa correctamente después de unos 3 segundos.	✔
Activación del sistema de ventilación	El sistema se activa correctamente después de unos 3 segundos.	✔
Desactivación del sistema de riego	El sistema se desactiva correctamente después de unos 3 segundos.	✔
Desactivación del sistema de ventilación	El sistema se desactiva correctamente después de unos 3 segundos.	✔
Conexión vía Ethernet	El dispositivo Arduino es visible en la red Ethernet	✔
Envío de paquetes UDP vía Ethernet	Los paquetes UDP son recibidos en el PC correctamente.	✔
Recepción de paquetes UDP vía Ethernet	Los paquetes UDP son recibidos en Arduino correctamente.	✔

Tabla 76 - Batería de pruebas de Arduino

APLICACIÓN DE ESCRITORIO		
Prueba	**Resultado**	**Validado**
Mostrar datos recibidos	Los datos recibidos son mostrados correctamente en la interfaz de usuario.	✔
Dibujar gráficos de evolución temporal	Se muestra correctamente la representación gráfica de la evolución reciente de las variables monitorizadas.	✔
El usuario desplaza el cursor por el área de los gráficos de evolución temporal	Se muestra el valor del gráfico en el punto relativo a la posición en el eje X del cursor del ratón.	✔
Mostrar nivel de agua disponible	Se muestra la representación gráfica del depósito de agua junto a su nivel de llenado porcentual.	✔
Activación del sistema de ventilación	El sistema se activa correctamente en Arduino.	✔
Desactivación del sistema de riego	El sistema se desactiva correctamente en Arduino	✔
Desactivación del sistema de ventilación	El sistema se desactiva correctamente en Arduino.	✔
Conexión vía Ethernet	El dispositivo Arduino es accesible en la red Ethernet por la aplicación de escritorio.	✔
Envío de paquetes UDP vía Ethernet	Los paquetes UDP son recibidos en el dispositivo Arduino.	✔
Recepción de paquetes UDP vía Ethernet	Los paquetes UDP son recibidos en la aplicación de escritorio.	✔
Mostrar condiciones actuales del invernadero.	Se muestran en la interfaz de usuario el estado de las condiciones actuales del invernadero mediante texto e imágenes representativas. Incluye mensajes de alerta.	✔
Mostrar modo de funcionamiento de los sistemas de riego y ventilación	El modo de funcionamiento actual de los sistemas de riego y ventilación es representado correctamente en los botones de control de cada sistema.	✔
Mostrar la actividad o inactividad de los sistemas de riego y ventilación	La actividad o inactividad de los sistemas de riego y ventilación es mostrada en la interfaz del usuario mediante animaciones de imágenes.	✔
Cambiar los rangos aceptables de las variables monitorizadas	Los valores máximos y mínimos relativos a cada variable son alterados por el usuario mediante los controles dispuestos en la interfaz.	✔
Mostrar los rangos aceptables de las variables monitorizadas	Se muestran en pantalla los valores máximos y mínimos permitidos de las variables monitorizadas.	✔
Mostrar el intervalo de actualización de los gráficos de evolución de las variables	Se muestra en la interfaz el valor empleado en el intervalo de actualización de los gráficos de evolución.	✔
Modificar el intervalo de actualización de los gráficos de evolución de las variables	El intervalo es modificado por el usuario a su voluntad. Las unidades son ajustadas correctamente al alcanzar una cantidad que supone el incremento de la unidad de tiempo superior. Por ejemplo, 61 segundos = 1 minuto y 1 segundo.	✔
Cambiar el modo de control de los sistemas de riego y ventilación	El sistema afectado cambia su modo de control con cada pulsación, rotando entre ON, OFF y AUTO. Las órdenes relativas al nuevo estado son enviadas y ejecutadas en el dispositivo Arduino.	✔
Mostrar fecha y hora de la última actualización de los gráficos evolutivos	Se muestra en la interfaz de usuario la fecha y hora en la que se actualizaron los gráficos por última vez	✔

Tabla 77 - Batería de pruebas de la aplicación de escritorio (I)

APLICACIÓN DE ESCRITORIO		
Prueba	**Resultado**	**Validado**
Recepción de órdenes procedentes de la aplicación Android	Los nuevos estados de los sistemas de riego y ventilación son aplicados. Se guarda la configuración y se actualizan las condiciones si es necesario.	✔
Notificación de actualización de la configuración a la aplicación Android	El mensaje de notificación de cambios en la configuración es enviado a la aplicación Android.	✔
Fallo en la conexión con el dispositivo Arduino	Se muestra la pantalla de configuración de los valores de conexión en la interfaz de usuario.	✔
Cambio en la configuración de los valores de conexión de red.	Los nuevos valores son aplicados y se reintenta la conexión con Arduino.	✔
Activación del sistema de ventilación de forma automática	Se comprueban las condiciones relativas al sistema de ventilación y el modo de control actual y se envían las órdenes correspondientes al dispositivo Arduino.	✔
Desactivación del sistema de ventilación de forma automática	Se comprueban las condiciones relativas al sistema de ventilación y el modo de control actual y se envían las órdenes correspondientes al dispositivo Arduino.	✔
Activación del sistema de riego de forma automática	Se comprueban las condiciones relativas al sistema de riego y el modo de control actual y se envían las órdenes correspondientes al dispositivo Arduino.	✔
Desactivación del sistema de riego de forma automática	Se comprueban las condiciones relativas al sistema de ventilación y el modo de control actual y se envían las órdenes correspondientes al dispositivo Arduino.	✔
Conexión con la base de datos del servidor XAMPP	La aplicación intenta realizar la conexión a la base de datos empleando el usuario y password establecidos. Si no es posible, se muestra un mensaje de error en debug.	✔
Carga e inicialización de la configuración de usuario	La aplicación carga los valores de configuración de rangos aceptables y modos de control desde el archivo de configuración del servidor. Si no es posible la carga, se emplea la configuración nula por defecto.	✔
Guardado de la configuración	La aplicación guarda los valores actuales de configuración en el archivo de configuración del servidor.	✔
Guardado de estado de condiciones y actividad de sistemas	La aplicación guarda el estado actual de las condiciones del invernadero y la actividad de los sistemas de riego y ventilación en el archivo correspondiente del servidor.	✔
Comprobación de condiciones	Las condiciones recibidas en las lecturas enviadas por Arduino son comparadas con los valores fijados en la configuración de usuario sobre los rangos aceptables y el modo de funcionamiento de los sistemas. Si se detectan cambios, se activan/desactivan los sistemas de riego y ventilación y se informa al usuario a través de la interfaz. Con cada cambio se guardan la configuración, el estado actual de los sistemas y las condiciones actuales del invernadero.	✔
Conexión con el servidor XAMPP	Si no es posible la conexión con el servidor XAMPP, la aplicación no se inicia.	✔

Tabla 78 - Batería de pruebas de la aplicación de escritorio (II)

APLICACIÓN PARA DISPOSITIVOS ANDROID		
Prueba	**Resultado**	**Validado**
Mostrar datos recibidos	Los datos recibidos son mostrados correctamente en la interfaz de usuario.	✔
Mostrar menú de opciones	Se muestra el menú de opciones con los botones asociados cuando el usuario pulsa el botón correspondiente	✔
Mostrar nivel de agua disponible	Se muestra la representación gráfica del depósito de agua junto a su nivel de llenado porcentual.	✔
Activación del sistema de ventilación	La orden de activación es enviada a la aplicación de escritorio y se ejecuta en el dispositivo Arduino.	✔
Activación del sistema de riego	La orden de activación es enviada a la aplicación de escritorio y se ejecuta en el dispositivo Arduino.	✔
Desactivación del sistema de riego	La orden de desactivación es enviada a la aplicación de escritorio y se ejecuta en el dispositivo Arduino.	✔
Desactivación del sistema de ventilación	La orden de desactivación es enviada a la aplicación de escritorio y se ejecuta en el dispositivo Arduino.	✔
Conexión con aplicación de escritorio	La aplicación de escritorio es accesible.	✔
Envío de paquetes UDP vía Ethernet	Los paquetes UDP son recibidos en la aplicación de escritorio	✔
Recepción de paquetes UDP vía Ethernet	Los paquetes UDP son recibidos en la aplicación Android.	✔
Mostrar condiciones actuales del invernadero.	Se muestran en la interfaz de usuario el estado de las condiciones actuales del invernadero mediante texto e imágenes representativas. Incluye mensajes de alerta.	✔
Mostrar modo de funcionamiento de los sistemas de riego y ventilación	El modo de funcionamiento actual de los sistemas de riego y ventilación es representado correctamente en los botones de control de cada sistema.	✔
Mostrar la actividad o inactividad de los sistemas de riego y ventilación	La actividad o inactividad de los sistemas de riego y ventilación es mostrada en la interfaz del usuario mediante animaciones de imágenes.	✔
Cambiar los rangos aceptables de las variables monitorizadas	Los valores máximos y mínimos relativos a cada variable son alterados por el usuario mediante los controles táctiles dispuestos en la interfaz.	✔
Mostrar los rangos aceptables de las variables monitorizadas	Se muestran en pantalla los valores máximos y mínimos permitidos de las variables monitorizadas.	✔
Cambiar el modo de control de los sistemas de riego y ventilación	El sistema afectado cambia su modo de control con cada pulsación, rotando entre ON, OFF y AUTO. Las órdenes relativas al nuevo estado son enviadas a la aplicación de escritorio y ejecutadas en el dispositivo Arduino.	✔

Tabla 79 - Batería de pruebas de la aplicación Android (I)

APLICACIÓN PARA DISPOSITIVOS ANDROID		
Prueba	**Resultado**	**Validado**
Recepción del mensaje de recarga de configuración	La configuración de usuario es recargada desde el archivo de configuración.	✔
Notificación de actualización de la configuración a la aplicación de escritorio	El aviso de recarga de configuración es enviado a la aplicación de escritorio.	✔
Fallo en la conexión con la aplicación de escritorio	Se muestra la pantalla de información con los valores de conexión empleados.	✔
Activación del sistema de ventilación de forma automática	Se comprueban las condiciones relativas al sistema de ventilación y el modo de control actual y se envían las órdenes correspondientes a la aplicación de escritorio.	✔
Desactivación del sistema de ventilación de forma automática	Se comprueban las condiciones relativas al sistema de ventilación y el modo de control actual y se envían las órdenes correspondientes a la aplicación de escritorio.	✔
Activación del sistema de riego de forma automática	Se comprueban las condiciones relativas al sistema de riego y el modo de control actual y se envían las órdenes correspondientes a la aplicación de escritorio.	✔
Desactivación del sistema de riego de forma automática	Se comprueban las condiciones relativas al sistema de riego y el modo de control actual y se envían las órdenes correspondientes a la aplicación de escritorio.	✔
Carga e inicialización de la configuración de usuario	La aplicación carga los valores de configuración de rangos aceptables y modos de control desde el archivo de configuración del servidor. Si no es posible la carga, se emplea la configuración nula por defecto.	✔
Guardado de la configuración	La aplicación guarda los valores actuales de configuración en el archivo de configuración del servidor.	✔
Guardado de estado de condiciones y actividad de sistemas	La aplicación guarda el estado actual de las condiciones del invernadero y la actividad de los sistemas de riego y ventilación en el archivo correspondiente del servidor.	✔
Comprobación de condiciones	Las condiciones recibidas en las lecturas reenviadas por la aplicación de escritorio son comparadas con los valores fijados en la configuración de usuario sobre los rangos aceptables y el modo de funcionamiento de los sistemas. Si se detectan cambios, se activan/desactivan los sistemas de riego y ventilación y se informa al usuario a través de la interfaz. Con cada cambio se guardan la configuración, el estado actual de los sistemas y las condiciones actuales del invernadero.	✔
Conexión con el servidor XAMPP	Si no es posible la conexión con el servidor XAMPP, la aplicación no se inicia.	✔

Tabla 80 - Batería de pruebas de la aplicación Android (II)

PÁGINA WEB		
Prueba	**Resultado**	**Validado**
Prueba de menú	Los botones mostrados en el menú de la página web enlazan a la página web correspondiente. El menú permanece en una posición estática aunque el contenido de la sección mostrada no pueda ser visualizado completamente sin hacer scroll.	✔
Mostrar de icono de alerta en el menú	El icono de alerta aparece sobre la sección *Estado actual* de la página web si se produce una situación de alerta obtenida del archivo de condiciones actuales.	✔
Generación de la sección *Últimas lecturas*	La sección *Últimas lecturas* es generada correctamente, incluyendo los sketches de consulta	✔
Conexión con el servicio MySQL no disponible	La página web muestra el log de errores generado.	✔
Archivo de configuración no disponible	La página web muestra el log de errores generado	✔
No existen registros en la base de datos	Se genera un mensaje de error si se produce una división por cero o excepción similar.	✔
Desplazamiento del cursor del ratón sobre los sketches de consulta	Se muestra una línea vertical en la posición actual del cursor del ratón, indicando el valor puntual del gráfico.	✔
Cambio en el valor de número de registros a mostrar	El nuevo valor del rango a utilizar en los sketches es recogido por la página web y se genera una nueva consulta que modifica la información presentada.	✔
Generación de la sección *Estadísticas*	La sección Estadísticas muestra el sketch utilizado para la visualización de las variables elegidas por el usuario con el contenido vacío.	✔
Selección de variables a mostrar en la sección *Estadísticas*	La página web recopila las casillas marcadas y el rango temporal a utilizar y construye una consulta para obtener los datos requeridos de la base de datos por medio del sketch. Una vez obtenidos los datos, se dibujan los gráficos correspondientes a la selección del usuario.	✔
Generación de la sección *Estado actual*	La sección *Estado actual* es generada correctamente a partir de la información disponible en la tabla *actual* de la base de datos y los archivos de configuración y condiciones actuales del invernadero. La información mostrada se actualiza correctamente cada 30 segundos.	✔
Generación de la sección *Consumo de agua*	La sección *Consumo de agua* es generada por medio de la recopilación de datos en la tabla *lecturas* de la base de datos.	✔
Generación de la sección *Temperaturas*	La sección *Temperaturas* es generada por medio de la recopilación de datos en la tabla *lecturas* de la base de datos.	✔
Mostrar la sección *Sobre Hortduino*	La sección *Sobre Hortduino* es mostrada correctamente.	✔
Acceso a la página web mediante un navegador que no soporta contenido de HTML5	La página web no se muestra correctamente. Se necesita actualizar el navegador del usuario a una versión mas reciente.	✘

Tabla 81 - Batería de pruebas de la Página Web

BLOQUE III

MANUAL DE USUARIO E INSTALACIÓN

Índice de contenido

Índice de ilustraciones

1 INTRODUCCIÓN

Este manual de usuario explica el uso e instalación de las diversas herramientas disponibles para el usuario del sistema Hortduino. En las cuatro secciones siguientes se detalla por separado el proceso de instalación y su uso con imágenes y textos descriptivos.

¿Qué es Hortduino?

Hortduino es un sistema de gestión y control remoto de invernadero. El término "Hortduino" viene de la unión de las palabras *hortelano* y *Arduino*, la popular plataforma de open-hardware de bajo coste. Este sistema cuenta con varias herramientas que ayudan al usuario a mantener las condiciones óptimas del crecimiento de la cosecha y recogen datos útiles para analizarlos y extraer conclusiones sobre el resultado obtenido.

¿Qué puede hacer Hortduino?

El sistema Hortduino es capaz de automatizar tareas como el riego o la ventilación del invernadero en base a los parámetros fijados por el usuario sobre las condiciones supervisadas. También ofrece información recibida desde los sensores instalados en tiempo casi real y de manera continuada, por lo que el usuario está informado en todo momento de las condiciones presentes del invernadero. Esta información se puede recibir en cualquier ubicación si el usuario emplea la aplicación Android del sistema para, por ejemplo, recibir la información y controlar el invernadero desde un smartphone que cuente con el sistema operativo Android, mientras se encuentra lejos del invernadero.

Si se desea acceder a los registros pasados de la información generada por Hortduino, la página web pone a disposición del usuario una serie de secciones individuales en la que se presentan la información obtenida de los sensores en gráficos informativos que el usuario puede variar para observar la evolución en el tiempo de las condiciones del invernadero.

¿Qué requisitos necesita el sistema Hortduino?

Se ha tratado de que los costes asociados a la puesta en marcha de este sistema sean asequibles, incluyendo en la medida de lo posible herramientas software con licencias gratuitas y dispositivos hardware económicos.

Los requisitos específicos de hardware y software para el funcionamiento del sistema Hortduino son mostrados en las tablas respectivas a continuación.

2 REQUISITOS MÍNIMOS

Requisitos de hardware

Instalación en el invernadero	Recursos asociados
Arduino MEGA 2560	Dispositivo Arduino
Sensor de temperatura exterior	National Semiconductor LM35
Sensor de humedad ambiental y temperatura interior	DHT11 Sensor
Sensor de luminosidad	Fotorresistencia
Sensor de humedad del suelo	Sensor de yeso (construcción propia)
Sensor de nivel de agua	Seed Ultrasonic Sensor
Activadores de los sistemas	Relés de 5V (x2)
Sistema de ventilación	Ventilador 12 V
Sistema de riego	Bomba de agua 12V

Dispositivos	Recursos asociados
PC Servidor XAMPP	Procesador de al menos 500 Mhz 128 MB Memoria RAM (256 MB recomendados) 300 MB de espacio libre en el disco duro
PC Cliente	128 MB Memoria RAM (256 MB recomendados) 140 MB de espacio libre en el disco duro
Dispositivo Android	Versión de Android 2.3.6 o superior Pantalla con resolución 640 x 480 píxeles 2 MB de espacio libre en almacenamiento interno Acceso Wi-Fi o equivalente

Tabla 1 - Requisitos de hardware

Requisitos de software

Dispositivos	Recursos asociados
PC Servidor XAMPP	Sistema operativo Windows, Linux o OSX Acceso a red local/Internet
PC Cliente	Sistema operativo Windows o Linux Java Runtime Environment versión 7 o superior Acceso a red local/Internet
Dispositivo Android	Android OS 2.3.6 o superior Acceso a red local/Internet

Tabla 2 - Requisitos de software

3 APLICACIÓN DE ESCRITORIO

Para ejecutar la aplicación de escritorio, primero se ha de comprobar que el servidor XAMPP se encuentra en funcionamiento y es accesible. Si se puede acceder a la página web de Hortduino, todo indica que el servidor trabaja sin problemas. Una vez comprobado lo anterior, solamente es necesario hacer doble-click en el icono de la aplicación para que ésta se inicie.

Si no es posible la comunicación con el dispositivo Arduino, se mostrará una pantalla similar a esta:

Ilustración 1 - Problemas de conexión con Arduino

En la pantalla anterior se muestra los datos de configuración de red que se está utilizando actualmente para conectar con Arduino. Se puede modificar esta configuración pulsando en el botón *CONFIGURAR CONEXION* que aparece en la parte inferior de la pantalla.

Para cambiar la dirección IP se debe escribir la nueva dirección IP a utilizar en la conexión y pulsar Intro para almacenar los datos. La misma operación se aplica para el puerto. En el caso de que no se pueda conectar por red con el dispositivo Arduino, es aconsejable comprobar la *Guía de resolución de problemas* incluida en este documento para establecer la causa que provoca esta ausencia de conexión.

Si todo funciona correctamente, la aplicación de escritorio mostrará en su ventana una interfaz de control y supervisión similar a la de la ilustración siguiente:

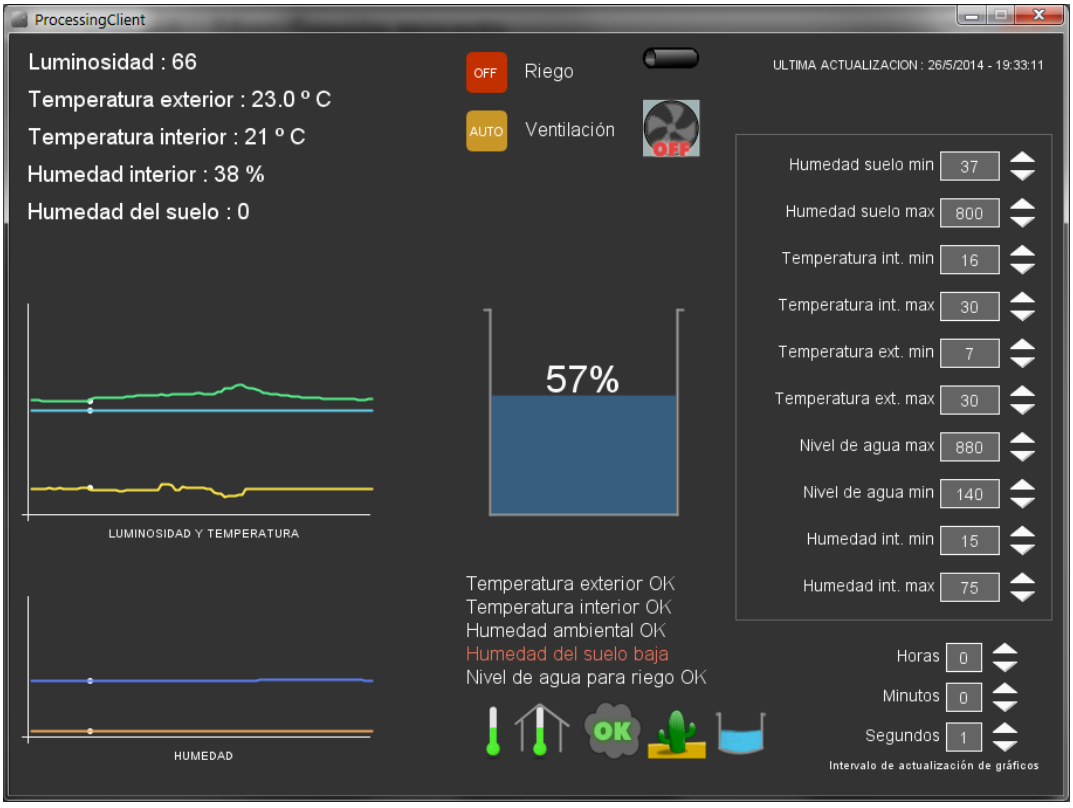


Ilustración 2 - Ventana de la aplicación de escritorio

Aquí, el usuario tiene a su disposición la información sobre los diversos aspectos del invernadero y los controles para establecer las condiciones aceptables y los modos de control de los sistemas de riego y ventilación. Los diferentes elementos que componen la interfaz son explicados en detalle a continuación.

3.1 Última lectura recibida de los sensores

En la zona superior izquierda de la ventana se muestra la información mas reciente recibida desde los sensores. Estos datos se reciben prácticamente en tiempo real desde los sensores conectados al hardware Arduino, con una frecuencia aproximada de 1.5 segundos entre actualizaciones.

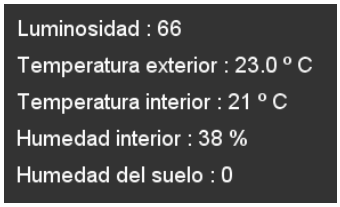


Ilustración 3 - Ultima lectura recibida desde los sensores

3.2 Gráficos de evolución reciente de las variables

Debajo de la información mas reciente se dibujan las gráficas correspondientes a los valores de luminosidad y temperaturas y a los valores de la humedad ambiental y del suelo. Estas dos gráficas no tienen escala definida y se utilizan principalmente para observar las oscilaciones de las variables monitorizadas en un espacio de tiempo que el usuario puede definir utilizando los controles situados en la parte inferior derecha de la ventana.

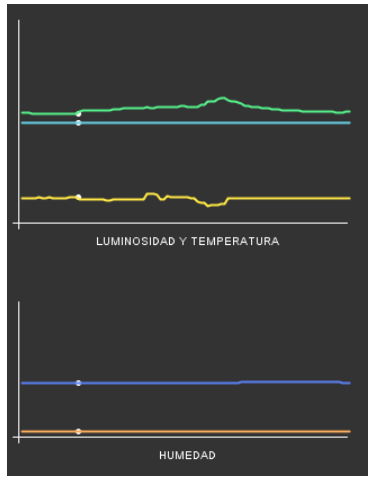


Ilustración 4 - Gráficos de las variables

Pueden resultar de utilidad para determinar el correcto funcionamiento de los sensores y detectar un posible fallo si se observan variaciones excesivas (picos) o lecturas anómalas, lo que indicaría un posible mal funcionamiento de hardware.

Al posicionar el cursor del ratón sobre el área de los gráficos se muestra el valor representado en ese punto de ambos gráficos. Cuando el trazado de los gráficos alcanza el límite derecho del eje X, se redibujan comenzando de nuevo en la posición de origen del eje X y se muestra un punto de color blanco sobre el gráfico para indicar la posición de la última actualización.

Las variables representadas en los gráficos son la temperatura exterior (verde), temperatura interior (azul claro), luminosidad (amarillo), humedad ambiental (azul oscuro) y humedad del suelo (naranja).

3.3 Controles y estados de los sistemas de riego y ventilación

Esta parte de la interfaz de usuario es la encargada de indicar el estado de los sistemas de riego y ventilación mediante dos pares de iconos. Los situados a la izquierda de la imagen actúan a modo de botón accionable por el usuario y alternan entre los estados ON, OFF y AUTO con los colores verde, rojo y ámbar respectivamente, estableciendo las órdenes a enviar al Arduino relativas a cada sistema. Por otra parte, los iconos situados a la derecha indican el estado actual del riego y la ventilación, mostrando animaciones si los sistemas están activos en ese momento.

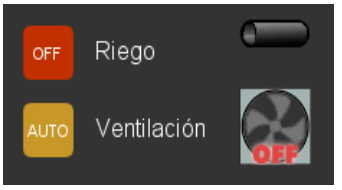


Ilustración 5 - Controles de riego y ventilación

En el modo automático, el sistema de ventilación responde automáticamente a las condiciones impuestas sobre humedad ambiental y temperatura interior mientras que el sistema de riego controla la humedad del suelo. Las condiciones de activación de los sistemas de riego y ventilación dependen de los valores fijados por el usuario mediante los selectores.

3.4 Condiciones actuales del invernadero

La información referente a las condiciones actuales del invernadero es mostrada mediante mensajes de texto e imágenes descriptivas del estado actual de cada condición. Si se produce una alerta en el estado de las condiciones, el texto y la imagen mostrada cambian para informar al usuario de que se ha producido una situación que requiere su atención.

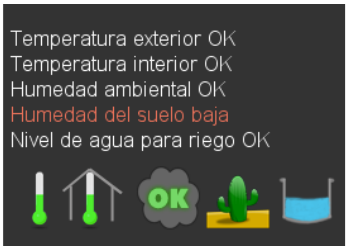


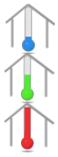
Ilustración 6 - Información de condiciones actuales

Estas condiciones son comprobadas cada vez que llega una nueva lectura desde los sensores, actualizando la información mostrada consecuentemente.

Las imágenes representativas de las condiciones son las siguientes:

- Temperatura exterior :
  - temperatura inferior al mínimo
  - temperatura nominal
  - temperatura superior al máximo

- Temperatura interior :
  - temperatura inferior al mínimo
  - temperatura nominal
  - temperatura superior al máximo



- Humedad ambiental :
  - humedad baja
  - humedad nominal
  - humedad elevada

- Humedad del suelo :
  - humedad baja
  - humedad nominal
  - humedad elevada

- Nivel de agua en depósito :
  - nivel de agua bajo
  - nivel de agua nominal
  - nivel de agua demasiado alto

3.5 Selectores de condiciones aceptables y actualización

Los selectores permiten al usuario cambiar el valor mínimo o máximo establecido para cada variable monitorizada. Para cambiar el valor contenido en la caja de texto, el usuario debe hacer click o dejar pulsado el botón izquierdo del ratón sobre el triángulo superior (si quiere aumentar el valor) o sobre el triángulo inferior (para disminuir el valor mostrado).

Ilustración 7 - Selectores de rangos aceptables

Si se desea cambiar la frecuencia de actualización de los gráficos mostrados sobre la evolución reciente de las variables, se procede de similar manera sobre el grupo de selectores situado debajo de los anteriores.

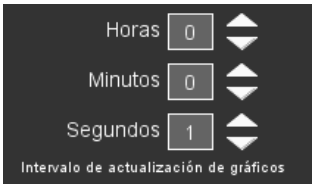


Ilustración 8 - Selectores de actualización

4 APLICACIÓN PARA DISPOSITIVOS ANDROID

Esta aplicación es una réplica reducida de la aplicación de escritorio que proporciona las funciones mas importantes de control e información, todo ello en una interfaz mas compacta. La aplicación para dispositivos Android se ejecuta pulsando sobre el icono disponible en el menú de aplicaciones del smartphone, representado por el icono en forma de H. En la ilustración siguiente se indica por medio de una flecha roja, para mayor claridad.

4.1 Inicio de la aplicación Android

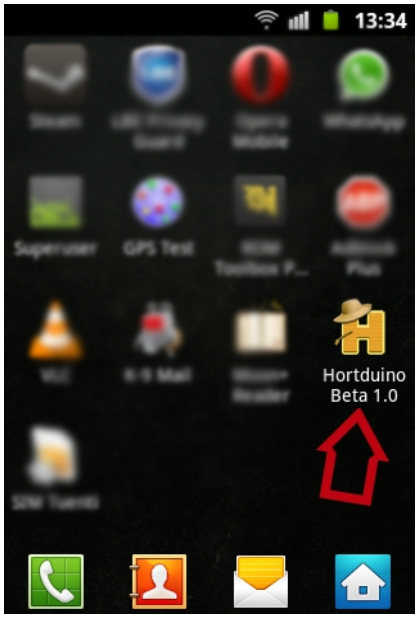


Ilustración 9 - Icono de la aplicación en Android

Es necesario contar con conectividad vía Wi-Fi o 3G para que la aplicación pueda iniciarse. Si no se dispone de esta conectividad, la aplicación fallará al iniciarse y se cerrará automáticamente.

Al igual que la aplicación de escritorio, esta aplicación depende para su inicio que el servidor XAMPP se encuentre disponible, pues comparte configuración con la aplicación de escritorio y necesita tener acceso a dicho archivo. También necesita de la ejecución de la aplicación de escritorio para recibir los datos enviados por Arduino y controlar los sistemas de riego y ventilación del invernadero. Si la conexión con esta última no es posible, se muestra una pantalla de información con los datos usados para la conexión como la mostrada en la figura 10.

4.2 Pantalla principal de la aplicación Android

Ilustración 10 - Fallo en la conexión

Si los requisitos de conexión antes mencionados se cumplen y la aplicación es iniciada por el usuario, la pantalla mostrada en el dispositivo Android será la siguiente:

Ilustración 11 - Pantalla principal de la app.

Los elementos en la interfaz de usuario que aparecen en la pantalla son los mismos que en la aplicación de escritorio, exceptuando los gráficos de evolución de las variables que no han sido incluidos.

De arriba a abajo, se presentan primero los datos reenviados por la aplicación de escritorio con la última información recibida desde los sensores del invernadero. A continuación aparece el estado de las condiciones actuales del invernadero, representado por las líneas de texto descriptivas y las imágenes asociadas a cada condición.

Seguidamente se muestran los botones que seleccionan el modo de control de los sistemas de riego y ventilación. Pulsando sobre cada botón se alterna el modo entre manual (ON, OFF) o automático (AUTO). Las imágenes que aparecen al lado de los botones muestran la actividad de los sistemas por medio de animaciones.

4.3 Menú de la aplicación

En la parte inferior de la pantalla aparece el botón para acceder al menú de opciones, el cual muestra una pantalla similar a esta:

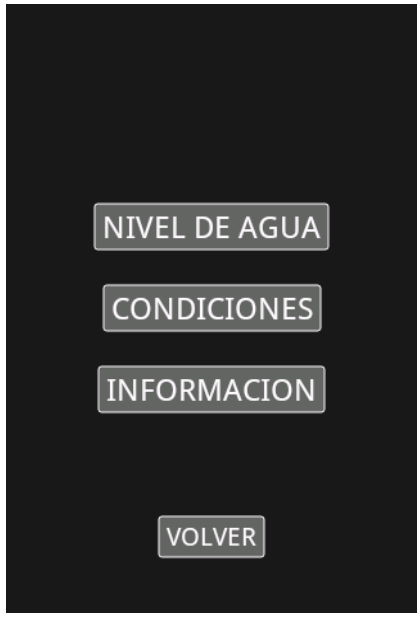


Ilustración 12 - Menú de la aplicación

IMPORTANTE! : Para retornar a la pantalla anterior, ha de emplearse siempre el botón VOLVER que aparece en la parte inferior de la pantalla en todas las secciones del menú. Si se pulsa el botón de retorno en el smartphone ( ) la aplicación se cerrará.

4.4 Nivel de agua disponible en el depósito

Si se pulsa sobre el botón NIVEL DE AGUA, se accede a la representación gráfica del depósito de agua para riego que muestra el nivel porcentual actual. El color del relleno varía con el nivel actual del depósito para mayor claridad visual.

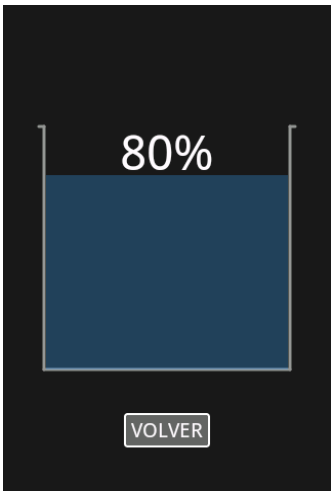


Ilustración 13 - Depósito al 80%

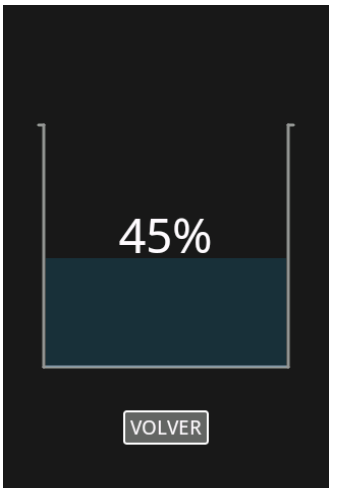


Ilustración 14 - Depósito al 45%

4.5 Cambiar los rangos aceptables de las condiciones

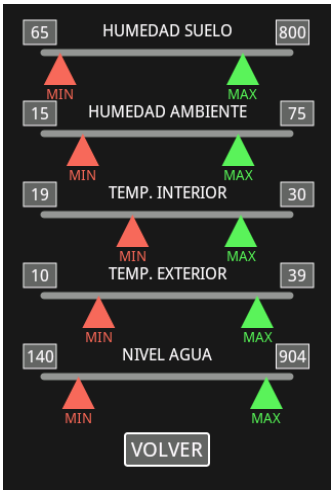


Ilustración 15 - Configuración de rangos aceptables

Para cambiar los valores máximos y mínimos aceptables en cada variable supervisada, se emplean un conjunto de selectores representados por triángulos sobre líneas horizontales. Cada triángulo representa el valor mínimo (color rojo) o máximo (color verde) que se permite sobre la variable en cuestión antes de que se genere una alerta o entre en funcionamiento el riego/ventilación automático.

Los valores mostrados en los extremos de la línea horizontal de cada variable son el valor mínimo (izquierda) y el valor máximo (derecha). Arrastrando el selector correspondiente se modifica el valor mostrado en cada caso.

La nueva configuración se hace efectiva al regresar a la pantalla principal de la aplicación.

4.6 Información de conexión

Esta sección ofrece datos acerca de la conexión establecida con la aplicación de escritorio, como la IP desde la que se reciben los datos y el numero de paquetes recibidos desde el inicio de la aplicación.

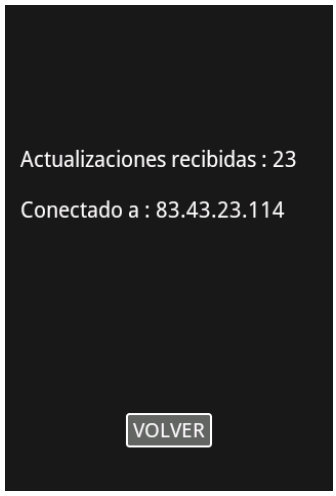


Ilustración 16 - Información de conexión

5 PÁGINA WEB

La página web de Hortduino presenta una estructura simple y sencilla, con un menú de opciones en la parte izquierda utilizado para elegir la sección a mostrar en la parte central de la página.

5.1 Sección *Últimas lecturas*

En esta sección se presentan las últimas lecturas recibidas mediante gráficos. Estos gráficos pueden variar la cantidad de datos representados si se cambia el valor de la lista desplegable, pudiendo escoger un intervalo de 1, 2, 4, 8 o 12 horas.

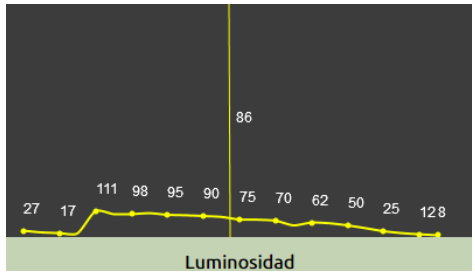


Ilustración 17 - Gráfico de Luminosidad

El usuario puede desplazar el cursor del ratón sobre el área del gráfico para visualizar el valor numérico asociado a la posición actual del cursor, señalado por una línea vertical del mismo color del gráfico, como puede verse en la ilustración 17.

5.2 Sección *Estadísticas*

La sección de estadísticas muestra un único espacio en el que son mostrados varias representaciones gráficas a la vez, correspondientes a las variables elegidas por el usuario empleando las casillas de selección. El intervalo temporal representado se puede variar mediante la lista desplegable que aparece debajo del área de selección de variables.

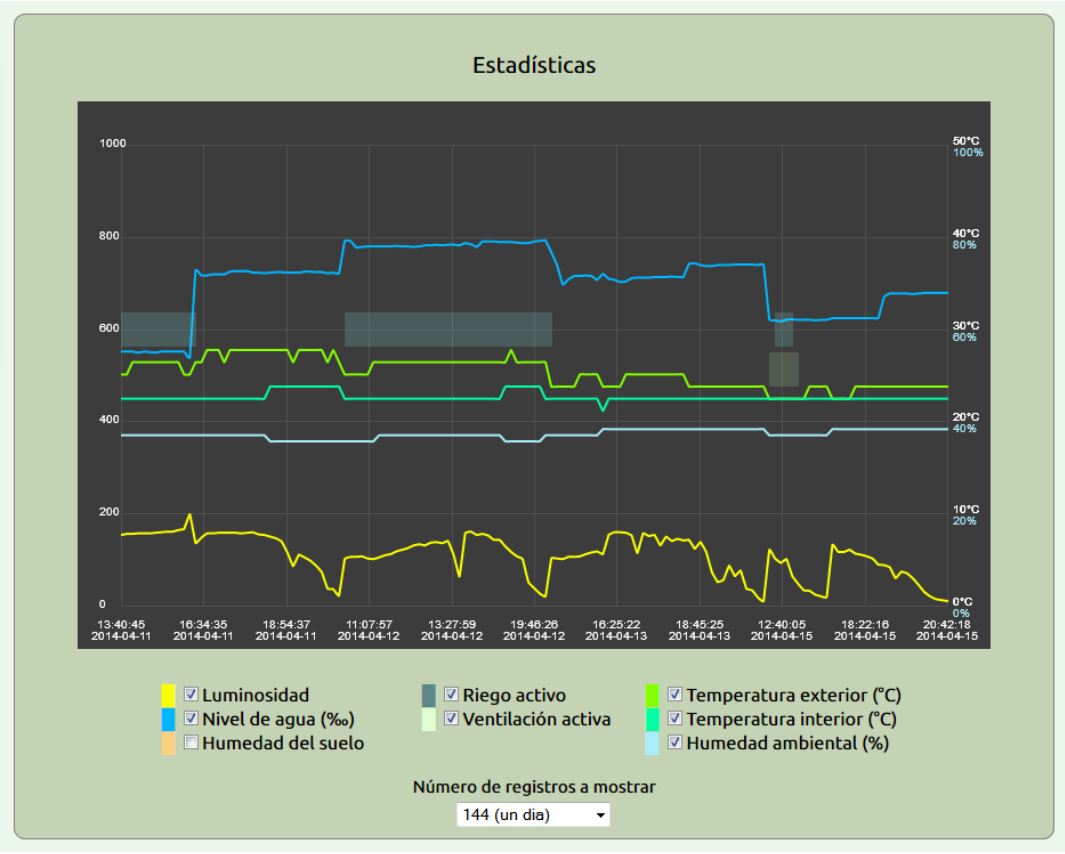


Ilustración 18 - Gráfico general de estadísticas

Las escalas mostradas a izquierda y derecha del eje Y están ligadas a la selección de las variables a graficar. De esta manera, si el usuario selecciona alguna variable del grupo izquierdo (*Luminosidad*, *Nivel de agua* o *Humedad del suelo*) se mostrará la escala relativa a esas variables en el eje Y izquierdo. En el caso de las variables del grupo derecho, se mostrará la escala de temperatura en el eje Y derecho si se selecciona al menos una de las variables de temperatura. Para el caso especial de la humedad ambiental, su escala se representa en el eje Y derecho con un color azul claro que acompaña al color del gráfico relativo a la humedad ambiental.

La actividad de los sistemas de riego y ventilación es representada en el gráfico de estadísticas mediante los rectángulos de color azul y verde claro que aparecen en la zona central del gráfico.

Los valores de la luminosidad y humedad del suelo no tienen asociados unidades de medida estándar sino que representan una medida perceptual sobre la condición que simbolizan. Así, la luminosidad puede variar desde el valor 0 (noche sin luna) hasta el valor 1000 (luz solar directa) con muchos valores intermedios y la humedad del suelo puede tomar valores desde 0 (ausencia total de humedad) hasta 1000 (saturación del sensor) según la conductividad relativa a la presencia de agua en el cuerpo del sensor.

5.3 Sección *Estado actual*

Esta sección es de carácter meramente informativo y recopila la última lectura recibida desde los sensores de Arduino, el estado de actividad y modo de funcionamiento de los sistemas de riego y ventilación y las condiciones actuales del invernadero. Si se produce una situación de alerta en alguna de las condiciones, el menú de la parte derecha mostrará un icono de advertencia ⚠ y aparecerá un mensaje en color rojo dentro de ésta sección indicando el problema encontrado.

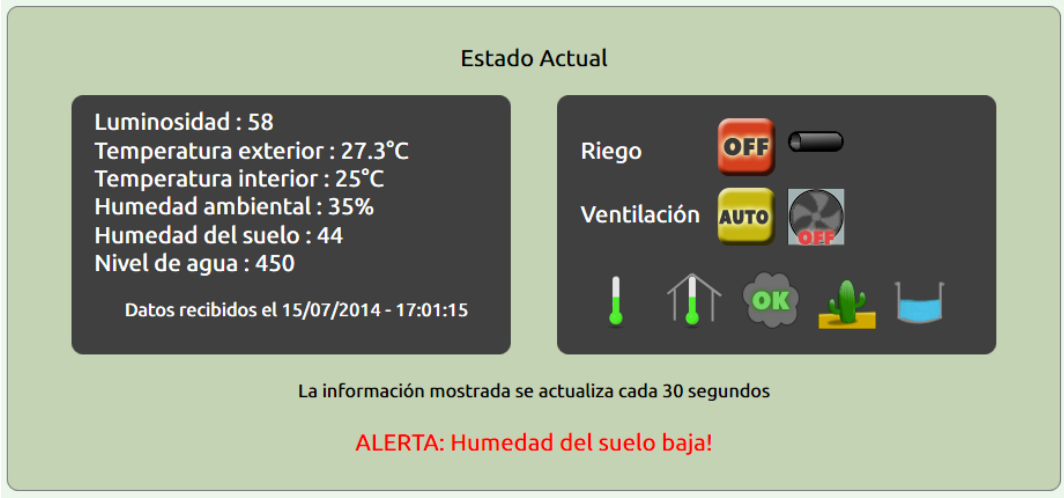


Ilustración 19 - Sección *Estado Actual* mostrando un mensaje de alerta

Las condiciones actuales están representadas gráficamente mediante las imágenes vistas en la página 8 de este manual. Si se mueve el cursor del ratón sobre las imágenes y se deja allí durante unos instantes, aparece la descripción y estado actual de la condición en forma de mensaje debajo del cursor del ratón.

5.4 Sección *Consumo de agua*

La sección consumo de agua muestra datos relevantes sobre el nivel de agua disponible en el depósito para riego, el consumo total de agua, consumo medio diario y la previsión de agua restante calculada con el consumo medio diario y el nivel de agua restante en el depósito.

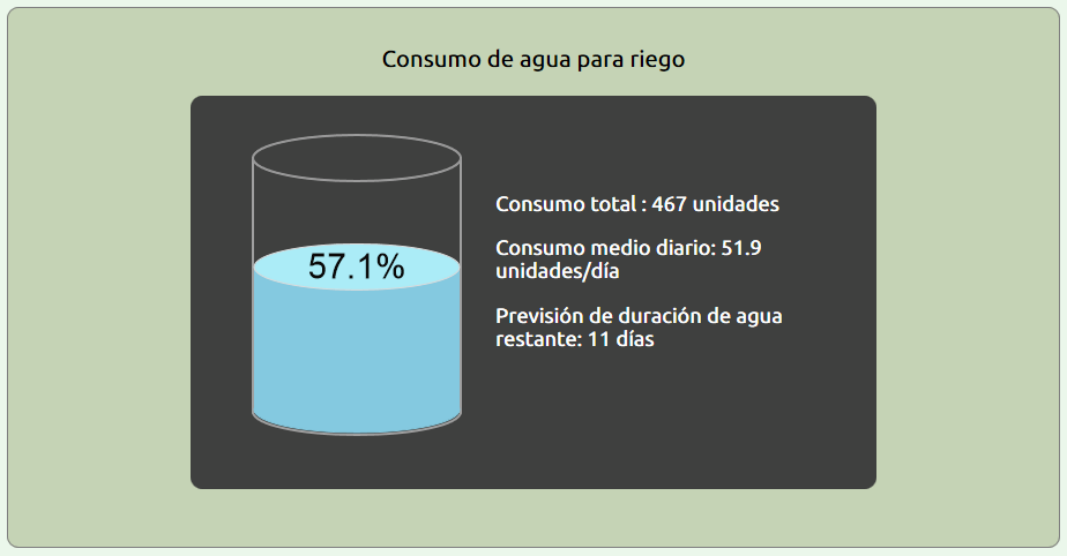


Ilustración 20 - Sección *Consumo de agua*

5.5 Sección *Temperaturas*

En esta sección se muestran los registros de las temperaturas medias, mínimas y máximas correspondientes al intervalo de días introducido en la caja de texto por el usuario.

Ilustración 21 - Sección *Temperaturas registradas*

5.6 Sección *Sobre Hortduino*

Aquí se explica brevemente el objetivo del sistema Hortduino, su funcionamiento y las diferentes partes que lo componen. Esta sección es la primera en visualizarse por el usuario cuando se accede a la web en el navegador.

6 GUÍA DE RESOLUCIÓN DE PROBLEMAS

Problema	**Solución**
La aplicación de escritorio no puede conectar con Arduino	Compruebe que el servidor XAMPP se encuentra en funcionamiento.
	Compruebe que la dirección IP y puerto de Arduino son correctos.
	Asegúrese que el dispositivo Arduino se encuentra en funcionamiento y tiene conectividad de red.
	Compruebe que el puerto 9631 UDP está abierto en la configuración del router de usuario para la dirección IP de Arduino.
La aplicación Android no puede conectar con la aplicación de escritorio	Asegúrese de que la aplicación de escritorio está funcionando correctamente.
	Compruebe que el servidor XAMPP se encuentra en funcionamiento.
	Compruebe que el puerto 9631 UDP está abierto en la configuración del router de la aplicación de escritorio.
La aplicación Android no se inicia en el smartphone	Asegúrese de que el dispositivo tiene conectividad de red (Wi-Fi o 3G)
La aplicación Android se cierra después de mostrar una pantalla en negro	Compruebe que el servidor XAMPP se encuentra en funcionamiento y es accesible de manera externa.
Los datos recibidos por la aplicación de escritorio no se guardan en la base de datos	Compruebe en el panel de control del servidor XAMPP que el servicio MySQL está en ejecución.
La página web muestra mensajes de errores	Compruebe en el panel de control del servidor XAMPP que el servicio MySQL está en ejecución.
	El archivo *condiciones.txt* está dañado o no es accesible en el servidor
	El archivo *config.txt* está dañado o no es accesible en el servidor
La configuración de la aplicación de escritorio o Android no se mantiene entre sesiones	Compruebe que el archivo *config.txt* en el directorio `/htdocs/hortduino` del servidor XAMPP no está protegido contra escritura.
Se muestran lecturas erróneas en la aplicación de escritorio o Android	Verifique que los sensores del invernadero están correctamente conectados y no existen interferencias.

7 MANUAL DE INSTALACIÓN

En esta sección se explica cómo instalar las diferentes herramientas software del sistema Hortduino. La instalación de los sensores y sistemas de riego y ventilación no se abordan en este manual y serán responsabilidad del usuario.

7.1 Programación de Arduino

Para cargar el programa en el dispositivo Arduino necesitaremos instalar el IDE y los drivers de Arduino en el PC. Este proceso está explicado en detalle dentro del tutorial *Getting Started with Arduino on Windows* de la página web de Arduino.

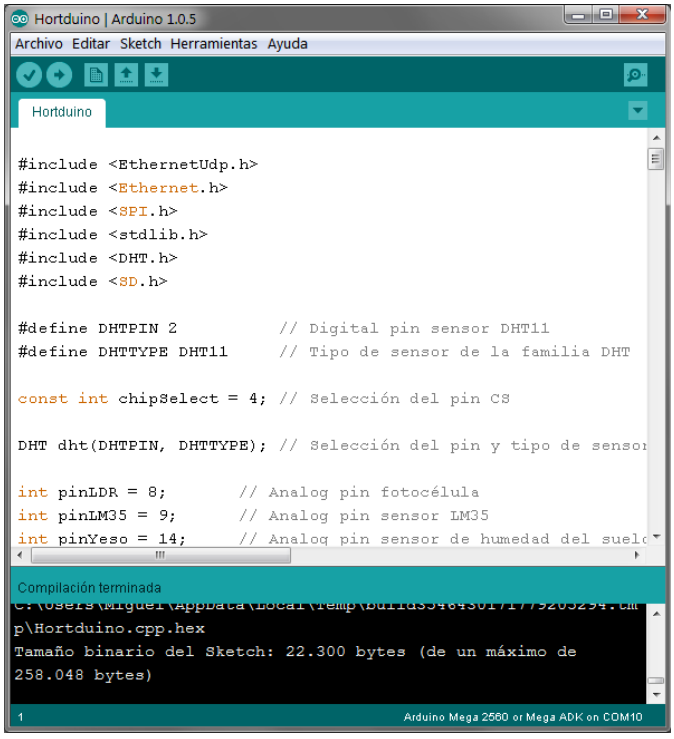


Ilustración 22 - Entorno de desarrollo de Arduino

Una vez instalado y configurado el entorno de desarrollo de Arduino, se procede a cargar el archivo `hortduino.ino` abierto en la aplicación Arduino en el dispositivo, empleando para ello el botón *Cargar* de la barra de botones de la aplicación.

7.2 Instalación de la aplicación de escritorio

La aplicación de escritorio requiere del entorno *Java Runtime Environment* versión 7 o superior, el cual puede ser descargado de https://www.java.com/es/download/. Es necesario que se mantenga el contenido y estructura del directorio de la aplicación para su correcto funcionamiento.

7.3 Instalación de la aplicación Android

Para instalar la aplicación en el dispositivo Android es necesario copiar el archivo de paquete de aplicación `Hortduino 1.0 beta.apk` en el almacenamiento interno y ejecutar su instalación con la utilidad *Instalador del paquete* de Android.

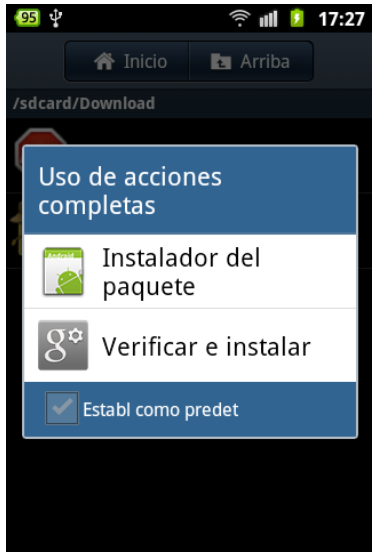


Ilustración 23 - Instalación en Android (I)

Se mostrará un mensaje de información sobre los permisos que requiere la aplicación sobre conexión a la red y el sistema de archivos. Pulse en *Instal.* y se finaliza la instalación.

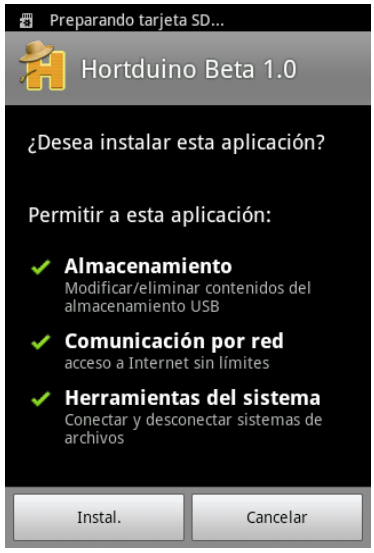


Ilustración 25 - Instalación en Android (II)

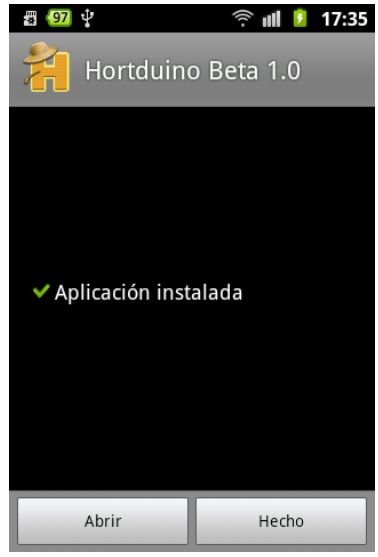


Ilustración 24 - Instalación en Android (III)

7.4 Instalación del servidor XAMPP

El paquete XAMPP se puede descargar desde la página web de Apache Friends en el siguiente enlace: https://www.apachefriends.org/es/index.html. Se puede elegir el paquete correspondiente al sistema operativo sobre el cual se instalará XAMPP. En este caso, se utiliza Windows.

Una vez que se ha descargado el instalador del paquete XAMPP, se procede a su instalación haciendo doble click sobre el paquete instalador, lo que mostrará una pantalla similar a esta:

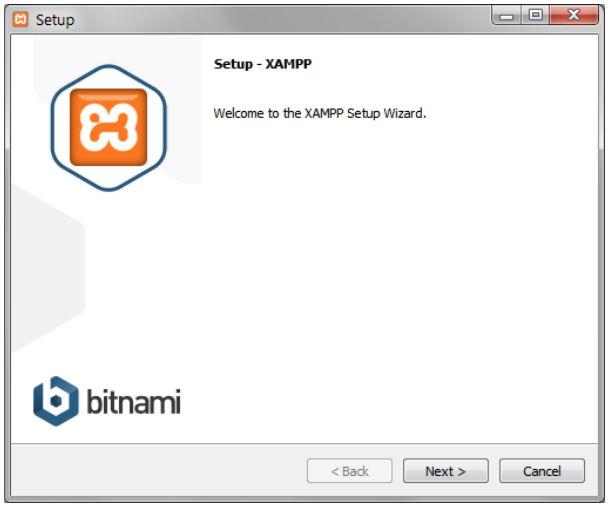


Ilustración 26 - Instalación del servidor XAMPP (I)

Para continuar con la instalación, se hace click en el botón Next en la parte inferior derecha de la ventana, como se muestra en la anterior ilustración.

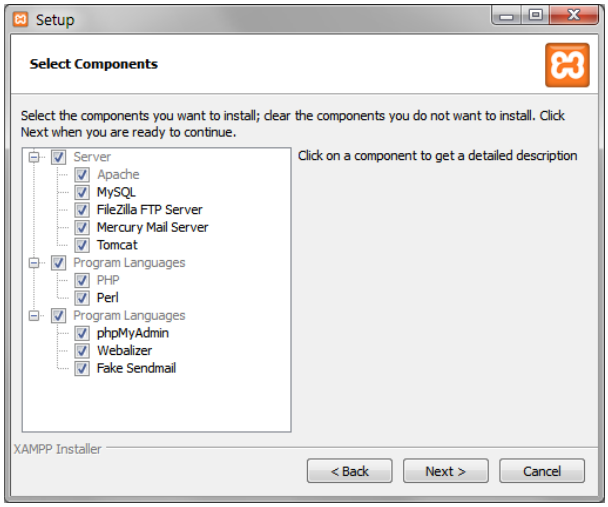


Ilustración 27 - Instalación del servidor XAMPP (II)

La instalación ofrece una lista de componentes a instalar en el PC. En este caso, solamente es necesario marcar las opciones **Apache**, **MySQL**, **PHP** y **phpMyAdmin**, ya que serán los componentes que se utilizarán en el funcionamiento del sistema Hortduino. Instalar la selección de componentes marcada por defecto, como se puede ver en la ilustración 27 de la página anterior, puede resultar útil si se desean utilizar las funciones de, por ejemplo, el servidor FTP FileZilla que viene incluido dentro de XAMPP.

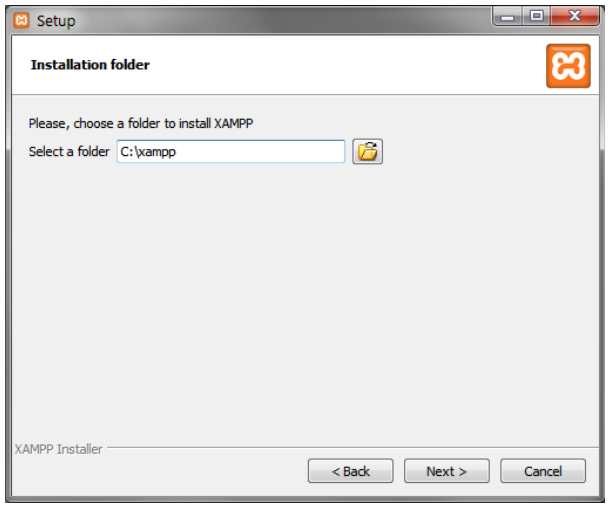


Ilustración 28 - Instalación del servidor XAMPP (III)

Después de seleccionar los componentes necesarios, aparece la selección del directorio donde se va a instalar el servidor XAMPP. En este caso se dejará el que se muestra por defecto, como se aprecia en la ilustración 28. Seguidamente, el proceso de copiado de archivos dará comienzo según se muestra en la ilustración siguiente:

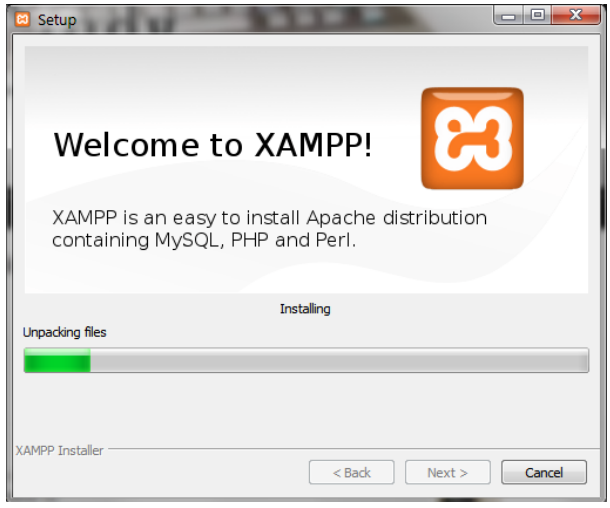


Ilustración 29 - Instalación del servidor XAMPP (IV)

7.5 Configuración de los servicios del servidor XAMPP

Una vez concluida la instalación del servidor XAMPP, se iniciará el panel de control desde el que se podrá controlar y configurar los distintos servicios que se hayan instalado. Los valores de configuración de los servicios Apache y MySQL para Hortduino son descritos mas adelante.

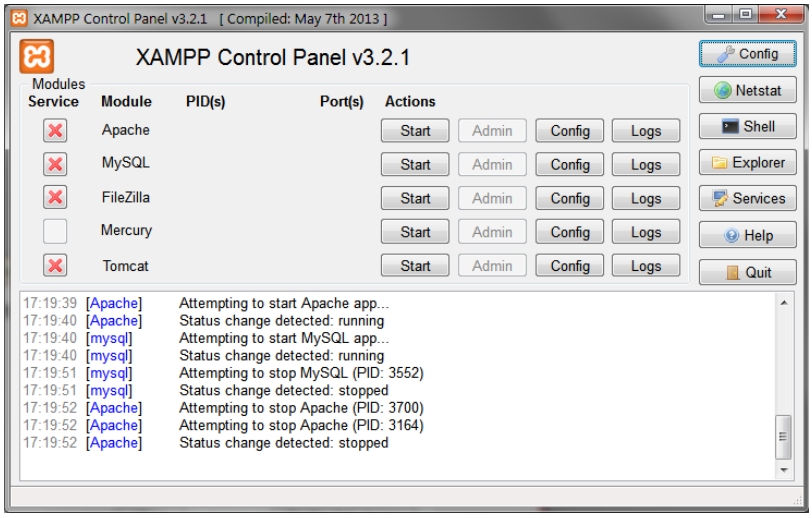


Ilustración 30 - Panel de control XAMPP

Desde el panel de control de XAMPP se puede iniciar o parar los servicios a voluntad del usuario, pero es conveniente que dichos servicios se inicien como procesos del sistema automáticamente sin intervención del usuario. Para ello, es necesario hacer click sobre el icono con una X roja al lado del nombre de los servicios Apache y MySQL y aceptar la modificación.

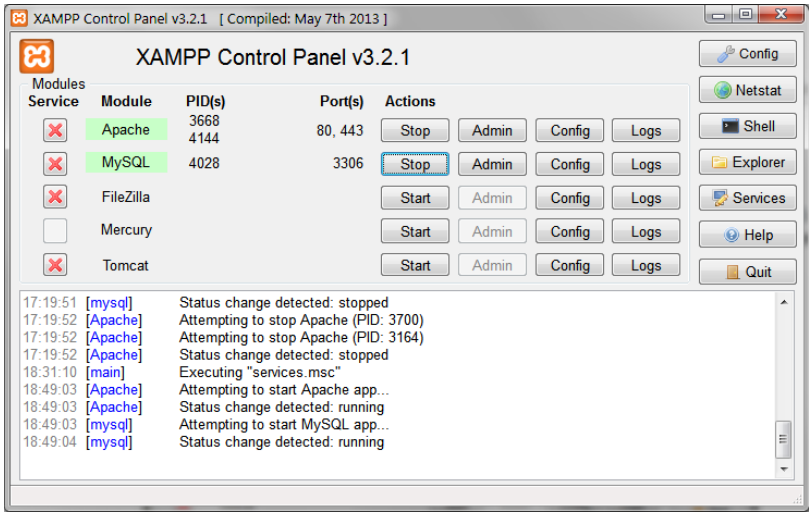


Ilustración 31 - Servicios Apache y MySQL en funcionamiento

Servicio Apache (servidor web)

La configuración del servidor Apache permite establecer varios parámetros relativos al control de listado de directorios, el directorio usado por la página web para almacenar el contenido y el directorio de usuarios (si hay mas de un usuario que guarde documentos en el directorio web).

Para configurar estos parámetros, es necesario pulsar sobre el botón *Config* en la línea del módulo Apache y seleccionar la opción **Apache (httpd.conf)** del menú desplegado, con lo que se abrirá el archivo de configuración en el editor de texto establecido por defecto en el sistema. En el caso particular de Hortduino, si la instalación de XAMPP se ha realizado en el directorio por defecto `C:\xampp\` solamente será necesario cambiar la sección **`#DocumentRoot`** y establecer las opciones de la siguiente forma:

```
DocumentRoot "C:/xampp/htdocs/hortduino"
<Directory "C:/xampp/htdocs/hortduino">
```

Si la instalación de XAMPP se realizó en otro directorio, hay que sustituir la ruta mostrada por la correspondiente al directorio de instalación. Una vez establecidos estos cambios, se guarda el fichero y se procede a copiar el directorio con el contenido de la página web de Hortduino en `C:/xampp/htdocs/`para que pueda ser mostrado en el navegador.

Intérprete PHP

Para configurar el intérprete de PHP se procede de igual manera que para la configuración del servicio Apache pero se elige la opción **PHP (php.ini)** en el menú que aparece después de hacer click sobre el botón *Config* de la sección Apache. En la sección **`;Paths and directories`** se debe cambiar las líneas siguientes:

```
include_path=".;C:\xampp\php\PEAR"
doc_root=
user_dir=
```

Servicio MySQL (motor de base de datos)

La configuración de MySQL se realiza a través de la interfaz web de administración *PhPMyAdmin*, a la que se accede mediante el botón *Config* de la sección correspondiente a MySQL en el panel de control de XAMPP. El password inicial del superusuario `root` está vacío por defecto, por lo que es necesario establecer uno nuevo inmediatamente por motivos de seguridad.

Ilustración 32 - Login en phpMyAdmin

Para utilizar la base de datos **Inver** con sus tablas asociadas que se adjunta en la documentación contenida en el CD, es necesario importarla a través de esta interfaz web. Esta operación se realiza desde el botón **Importar** en la parte superior de la página web PhPMyAdmin, seleccionando el archivo `inver.sql` en la ventana de selección desplegada desde el botón *Examinar* como puede observarse en la siguiente ilustración.

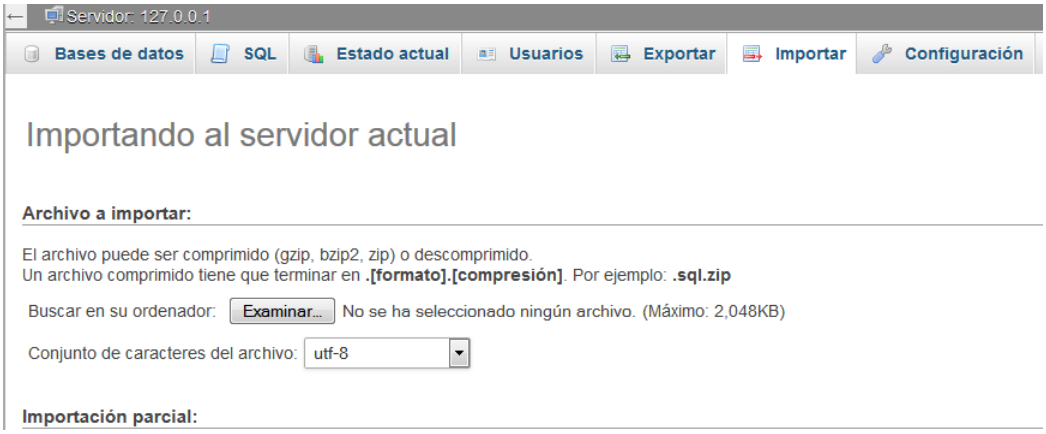


Ilustración 33 - Importar base de datos

Después de importar la base de datos, es necesario crear un usuario que tenga acceso a ella para realizar operaciones de consulta e inserción. Para esta operación, se selecciona la base de datos **Inver** en el menú situado en la parte izquierda de la página *PHPMyAdmin* y se hace click en el botón con la etiqueta **Privilegios**. Una vez dentro de esta opción, es necesario crear un usuario nuevo a través de la opción *Agregar usuario* con las siguiente configuración:

Información de usuario
Nombre: hortelano Contraseña: hortelano Privilegios globales: marcar todos

En la ilustración de la siguiente página se pueden apreciar las opciones de configuración del usuario *hortelano* para la base de datos *Inver* del servidor para mayor claridad.

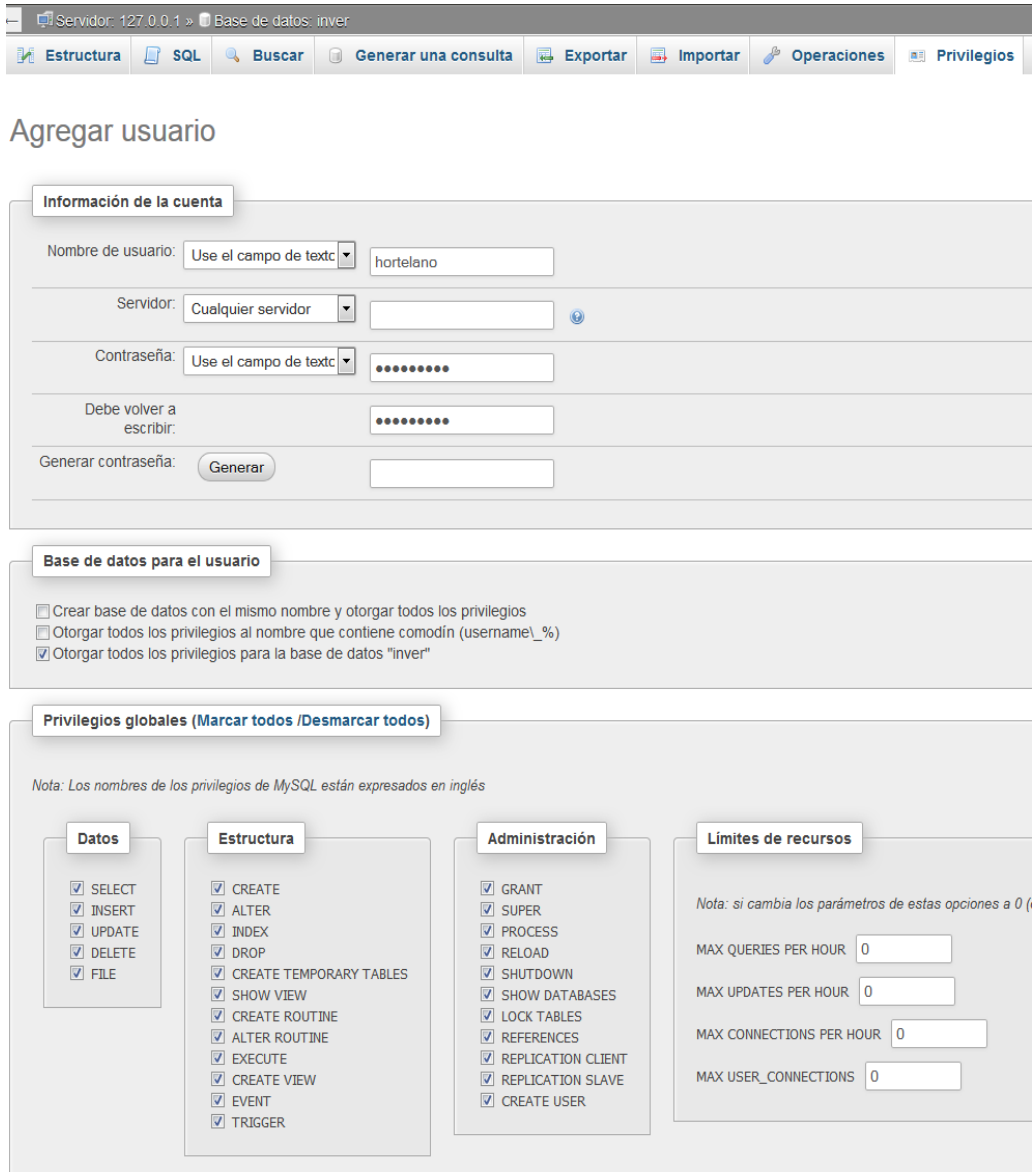


Ilustración 34 - Creación del usuario *hortelano* en la base de datos *Inver*

Este usuario es utilizado internamente en el código de la aplicación de escritorio y en la página web, por lo cual es de gran importancia que se añada a los permisos de la base de datos **Inver** para que se puedan llevar a cabo las tareas de inserción y consulta de registros.