|  |
| --- |
|  |
| Red Sensorial Inalámbrica para previos agrícola basada en Arduino |
|  |
| WSN v1 |

21 de Marzo de 2014

Autor: Gonzalo Muñoz Rojas;gmunoz10bio@hotmail.com

Red Sensorial Inalámbrica para previos agrícola basada en Arduino

WSN v1

Índice

[1. Introducción 3](#_Toc383116485)

[1.1. Propósito 3](#_Toc383116486)

[1.2. Descripción breve del problema 3](#_Toc383116487)

[2.1. Planificación temporal de actividades 4](#_Toc383116488)

[2.1.1. Informe General 4](#_Toc383116489)

[3. Análisis 4](#_Toc383116490)

[3.1. Contexto 4](#_Toc383116491)

[3.1.1. Descripción General 4](#_Toc383116492)

[3.1.2. Descripción de Clientes y Usuarios 5](#_Toc383116493)

[3.2. Especificación de Requerimientos 5](#_Toc383116494)

[3.2.1. Funciones del Sistema 5](#_Toc383116495)

[3.2.2. Atributos del Sistema 9](#_Toc383116496)

[3.2.3. Atributos por Función 10](#_Toc383116497)

[3.3. Actores 16](#_Toc383116498)

[3.4. Casos de uso 17](#_Toc383116499)

[3.4.1. Caso de uso esencial 17](#_Toc383116500)

[3.4.2. Diagrama de Caso de Uso 28](#_Toc383116501)

[3.4.3. Modelo Conceptual 32](#_Toc383116502)

[3.4.4. Diagrama de Secuencia o Colaboración 32](#_Toc383116503)

[3.4.5. Priorización 38](#_Toc383116504)

[3.5. Modelo de dominio 40](#_Toc383116505)

[3.5.1. Entidades Reconocidas 40](#_Toc383116506)

[3.5.2. Modelo de Dominio 41](#_Toc383116507)

[3.5.3. Matriz de Rastreabilidad 41](#_Toc383116508)

[4. Validación 44](#_Toc383116509)

[4.1. Prototipo de validación funcional 44](#_Toc383116510)

[5. Diseño 46](#_Toc383116511)

[5.1. Derivación del Modelo de Software 46](#_Toc383116512)

[5.1.1. Modelo de software inicial 46](#_Toc383116513)

[5.1.3. Diagrama de Clases 47](#_Toc383116514)

[5.1.4. Diagramas de Estados 49](#_Toc383116515)

[5.1.5 Patrones GRASP 51](#_Toc383116516)

[6. Implantación 52](#_Toc383116517)

[6.1. Código fuente completo (parcial) 52](#_Toc383116518)

[6.1.1 Clase Comando 52](#_Toc383116519)

[6.1.2. Clase Red 53](#_Toc383116520)

[6.1.3. Clase Comunicación 54](#_Toc383116521)

[6.1.4 Clase Almacenar SD 55](#_Toc383116522)

[6.1.5. Clase Sensor Nivel 56](#_Toc383116523)

[6.1.6. Clase Sensor Temperatura 56](#_Toc383116524)

[6.1.7. Clase Sensor Humedad 57](#_Toc383116525)

[6.1.8. Clase Sensor Flujo 57](#_Toc383116526)

[6.1.9. Clase Almacen 58](#_Toc383116527)

[6.2. Dependencias 59](#_Toc383116528)

[7. Anexos 60](#_Toc383116529)

[7.1. Glosario 60](#_Toc383116530)

[8. Bibliografía 62](#_Toc383116531)

# 1. Introducción

## Propósito

El presente documento plantea un prototipo desarrollo de software sobre la obtención de datos provenientes de sensores de forma remota, específicamente con un enfoque en el sector agrícola específicamente sobre plantaciones. Dicho proyecto va en la dirección que posee algunos proyectos como RCOM, sin embargo, se establecen diferencias en la transferencia de datos y el sistema informático involucrado. Va dirigido a personas con conocimiento informático y agrícola para aprovechar los datos entregados.

## Descripción breve del problema

Se define el problema principalmente como: Falta de un óptimo proceso para obtención de datos de lugares remotos y la implementación de una plataforma digital que permita su control y almacenamiento de los datos obtenidos. La solución a este problema se llevará a cabo mediante el desarrollo de un sistema para el control sobre placas de arduino, estableciendo comunicación con equipos remotos utilizando las herramientas de diseño y programación orientada a objetos como C++, para interfaz gráfica paquete como gtkmm y MySQL.

La mayor parte de la información será obtenida desde los nodos sensores o equipos remotos en base a peticiones que el usuario realice.

## 2.1. Planificación temporal de actividades

Ahora presentamos la planificación separada por entrega de Informe de Avance por sección. La planificación está sujeta a cambios debido al progreso que existe en proyecto.

### 2.1.1. Informe General

A continuación se describe la distribución de trabajo para el desarrollo de la primera fase del desarrollo del software correspondiente a la identificación del problema, presentación de una solución y la ejecución de trabajos, entre otros.

Figura 1: Distribución de trabajo para la segunda fase de desarrollo del software correspondiente a la Construcción base de dato, Análisis Servicio Web, Transferencia de datos, Conexión Ethernet arduino, Transferencia DE DATOS POR RED, PRUEBAS SISTEMA Y CONSTRUCCION GUI.

# 3. Análisis

A continuación se presentará el análisis del problema, junto a una descripción general, una identificación y caracterización de los clientes y usuarios, los requerimientos del sistema, actores, casos de uso y modelo de dominio del sistema para llegar a una solución.

## 3.1. Contexto

A continuación se describe el contexto en el cual estará enmarcada la utilización del software junto con los clientes y usuarios que interaccionarán con este.

### 3.1.1. Descripción General

WSN v1 está pensado como un software de automatización para la adquisición de datos desde el medio ambiente en algún previo agrícola u otro lugar de interés. La adquisición de datos desde el medio ambiente es de vital importancia para lograr entender el proceso maduración de frutas y el crecimiento de árboles, esto es, lograr desarrollar una planificación y estrategias para lograr obtener un producto de calidad. El problema es que hoy en día los datos de importancia para los previos agrícolas son obtenidos por medio de Servicios Meteorológicos que actúan en base a zonas bastas para poder establecer una predicción los más cercana a la realidad sobre las condiciones ambientales día a día que estarán presentes, además de tener centros meteorológicos los cuales pueden llegar a ser con un alto precio. Al estar dependiendo de registros medio ambientales que son obtenidos a partir de una zona amplia pueden llegar a generar una menor productividad en los previos agrícolas, además se debe considerar destinar recursos para movilizarse a los lugares en donde se encuentran estas plataformas incurriendo en gastos como tiempo de trabajo, combustible, planificación de la toma de muestras para obtener los datos de interés como su mantenimiento.

El software automatiza la adquisición de datos por medio de una interfaz intuitiva donde se presenta una ventana con opciones en cascada para obtener la información específica de un sensor en un lugar en concreto para así lograr obtener las condiciones actuales del terreno o plantación.

El sistema estará dirigido a usuarios que administren previos agrícolas u personal capaz de interpretar los datos, el usuario quien completa los pasos con la información de la solicitud.

Una vez ingresada la petición para la adquisición de datos será enviada al equipo elegido por medio de una red inalámbrica en donde el nodo sensor está a la espera de alguna orden, luego de enviada la orden esta es procesada por el nodo sensor y entrega una respuesta con los datos adquiridos en el momento. La hora de registro en la adquisición de datos es cuando estos son solicitados.

### 3.1.2. Descripción de Clientes y Usuarios

Cliente: El cliente de este software debe cumplir con requisitos de comprensión de los datos entregados por el sistema es decir el cliente puede ser un agrónomo como un entendido del tema. El Cliente será quien realizará las tareas administrativas sobre la información que se dispondrá sobre cada nodo sensor como también la actualización de la misma. Eventualmente el software podrá ser implementado en otras áreas que deseen automatizar la adquisición de datos sensores, en cuyo caso los clientes también tendrán que cumplir con requisitos descrito anteriormente.

Usuarios: Los usuarios del software serán agrónomos, técnicos agrícolas o usuario que sea entendido en el tema de adquisición de datos. Los usuarios serán los encargados de realizar las solicitudes de adquisición de datos en el sistema.

## 3.2. Especificación de Requerimientos

A continuación se realizará una descripción de los requerimientos del sistema, caracterizando las funciones, los atributos y la relación de ellos en el software.

### 

### 3.2.1. Funciones del Sistema

A continuación de describen y categorizan las funciones asociadas al funcionamiento del sistema. Esta sección tendrán grupos de funciones separados en: Control del sistema, Almacenar información, Comunicación del Sistema, Establecer Red, Configuración Módulos de Comunicación, Sensores, Servidor y Ventana.

|  |  |  |  |
| --- | --- | --- | --- |
| 1.0 | Control del Sistema | | |
| Ref. # | **Función** | **Descripción** | **Categoría** |
| R.1.1 | Setup | El sistema deberá permitir la configuración de las placas de manera automática | Oculta |
| R.1.2 | Loop | El sistema deberá permanecer en forma permanente a la escucha de órdenes de trabajo. | Evidente |
| R.1.3 | Confirmar Mensaje | El sistema deberá permitir la confirmación de un mensaje que sea enviado entre equipos | Oculta |
| R.1.4 | Control Sensor | El sistema deberá permitir realizar el ciclo de trabajo sobre un sensor específico. | Evidente |
| R.1.5 | Memoria Libre | El sistema deberá obtener la memoria ram libre de presente en el equipo. | Oculta |

Tabla 1: Lista de funciones del software para el control del sistema.

|  |  |  |  |
| --- | --- | --- | --- |
| 2.0 | Almacenar Información | | |
| Ref. # | **Función** | **Descripción** | **Categoría** |
| R.2.1 | Configurar Memoria | El sistema deberá permitir la configuración de la memoria para la utilización posterior. | Oculta |
| R.2.2 | Escribir en Memoria | El sistema deberá registrar los datos que sean solicitados. | Oculta |
| R.2.3 | Error en Memoria | El sistema deberá alertar sobre errores en el registro de datos. | Oculta |
| R.2.4 | Leer Memoria | El sistema deberá permitir la lectura de información desde la memoria | Oculta |
| R.2.5 | Crear | El sistema deberá permitir la creación de registros para almacenar datos | Oculta |
| R.2.6 | Liberar Memoria | El sistema deberá lograr liberar memoria que sea solicitada | Oculta |
| R.2.7 | Entregar Datos | El sistema deberá realizar la tarea de entrega de datos almacenados en memoria | Oculta |

Tabla 2: Lista de funciones del software para Almacenar información.

|  |  |  |  |
| --- | --- | --- | --- |
| 3.0 | Comunicación del Sistema | | |
| Ref. # | **Función** | **Descripción** | **Categoría** |
| R.3.1 | Enviar mensaje | El sistema deberá permitir enviar mensajes por medio de funciones integradas en Arduino. | Oculta |
| R.3.2 | Recibir Mensaje | El sistema deberá almacenar y mostrar el mensaje recibido por medio de funciones integradas | Oculta |
| R.3.3 | Borrar Mensaje | El sistema deberá permitir borrar el mensaje utilizado. (Requerimiento por memoria de Dispositivo) | Oculta |
| R.3.4 | Convertir Mensaje | El sistema deberá permitir la conversión del algún mensaje a numero para ahorrar espacio de memoria | Oculta |
| R.3.5 | Recuperar Mensaje | El sistema deberá permitir la obtención del mensaje recibido. | Evidente |
| R.3.6 | Limpiar Buffers | El sistema deberá permitir la limpieza de los buffer del dispositivo para no acumular datos basura. | Oculta |
| R.3.7 | Setear Mensaje | El sistema deberá permitir Setear el mensaje para su posterior utilización al enviar datos. | Oculta |

Tabla 3: Lista de funciones del software para la Comunicación del Sistema.

|  |  |  |  |
| --- | --- | --- | --- |
| 4.0 | Establecer Red | | |
| Ref. # | **Función** | **Descripción** | **Categoría** |
| R.4.1 | Iniciar Red | El sistema deberá realizar la configuración de la red para conectarse a un Servicio Web. | Oculta |
| R.4.2 | Leer mensaje | El sistema deberá permitir la lectura de un mensaje enviado por la conexión de red. | Oculta |
| R.4.3 | Eliminar Mensaje | El sistema deberá permitir borrar el mensaje utilizado. (Requerimiento por memoria de Dispositivo) | Oculta |
| R.4.4 | Procedimiento de Envío | El sistema deberá permitir asegurar el envió de un mensaje hacia el servidor | Oculta |
| R.4.5 | Envió por Red | El sistema deberá enviar un mensaje por la red establecida. | Oculta |
| R.4.6 | Error de Conexión | El sistema deberá alertar si el mensaje no ha logrado ser entrego al servidor | Oculta |
| R.4.7 | Enviar mensaje | El mensaje es enviado por esta función | Oculta |

Tabla 4: Lista de funciones del software para establecer red.

|  |  |  |  |
| --- | --- | --- | --- |
| 5.0 | Configuración Módulos de Comunicación | | |
| Ref. # | **Función** | **Descripción** | **Categoría** |
| R.5.1 | Modo Comando | El sistema deberá permitir el acceso a la configuración de los módulos de comunicación para dirigir las solicitudes de información. | Oculta |
| R.5.2 | Configurar parámetros | El sistema deberá lograr efectuar cambios sobre los módulos de comunicación para dirigir los mensajes | Oculta |
| R.5.3 | Salir Modo Comando | El sistema deberá salir del modo comando de los módulos de comunicación para grabar configuración correctamente. | Oculta |
| R.5.4 | Restablecer | El sistema deberá eliminar configuraciones previas del módulo de comunicación para dejarlo con configuración de fábrica | Oculta |
| R.5.5 | Explorar red | El sistema deberá explorar la red que los módulos de comunicación crean para obtener los equipos disponibles. | Oculta |
| R.5.6 | Obtener Equipos | El sistema deberá entregar una lista con los dispositivos actuales conectados a la red | Evidente |

Tabla 5: LISTA DE FUNCIONES DEL SOFTWARE PARA LA configuración módulos de comunicación.

|  |  |  |  |
| --- | --- | --- | --- |
| 6.0 | Sensores | | |
| Ref. # | **Función** | **Descripción** | **Categoría** |
| R.6.1 | Transformar Medida | El sistema deberá obtener un valor cuantitativo a partir de un valor analógico. | Oculta |
| R.6.2 | Obtener Medida | El sistema deberá entregar la medida solicitada a un sensor específico. | Oculta |

Tabla 6: Lista de funciones del software para sensores.

|  |  |  |  |
| --- | --- | --- | --- |
| 7.0 | Servidor | | |
| Ref. # | **Función** | **Descripción** | **Categoría** |
| R.7.1 | Abrir Servidor | El sistema establecerá una dirección ip en conjunto con una puerta de enlace como servidor | Evidente |
| R.7.2 | Manejar Conexiones | El sistema deberá permitir la entrada de conexiones nuevas en cualquier instante. | Oculta |
| R.7.3 | Función sql | El sistema logrará almacenar información en una base de datos | Oculta |
| R.7.4 | Función parsear | El sistema preparará la orden a ejecutar la función sql | Oculta |

Tabla 7 lista de funciones del software para el servidor.

|  |  |  |  |
| --- | --- | --- | --- |
| 8.0 | Ventana | | |
| Ref. # | **Función** | **Descripción** | **Categoría** |
| R.8.1 | Listar Puertos | El sistema listara los puertos donde se conecten equipos clasificado como nodo receptor | Evidente |
| R.8.2 | Listar Equipos | Véase R.5.6. | Evidente |
| R.8.3 | Listar Sensores | El sistema listara los sensores para un nodo en específico | Evidente |

Tabla 8 LISTA DE FUNCIONES DEL SOFTWARE PARA la ventana.

### 3.2.2. Atributos del Sistema

A continuación se presenta el listado de atributos del software, incluyendo detalles y limitaciones.

|  |  |  |
| --- | --- | --- |
| Ref. # | Atributo | Detalle y Limitación |
| Ref. 1 | Comunicación | El sistema ofrece un espectro en la banda de frecuencia de comunicación para la transferencia de datos manteniendo la integridad de los datos. La frecuencia será dependiente del hardware a utilizar. |
| Ref. 2 | Interfaz | El sistema posee una interfaz intuitiva con un diseño según el estándar de las Guías de Desarrollo de Gnome*.*  Los procesos desarrollados serán simples y claros, omitiendo procesos que sean complejos y engorrosos para el usuario. |
| Ref. 3 | Especifico | Los datos almacenados en memoria solo corresponderán a sensores identificados por una clave para lograr acceder a ellos. |
| Ref. 4 | Velocidad | El tiempo de respuesta en procesos de solicitud de información no deberá superar el minuto de espera. |
| Ref. 5 | Conectividad | El sistema permite la conectividad con una red Ethernet para la transferencia de datos por medio de una conexión LAN. |
| Ref.6 | Rendimiento | El sistema deberá liberar memoria que se requiera para logar obtener un rendimiento en el hardware de Arduino |

Tabla 9: Lista de atributos del software.

### 3.2.3. Atributos por Función

A continuación se presenta el listado de atributos por funcionalidad del software, especificando los atributos que debe tener cada función según corresponda. El listado también está separado en los grupos: Control del sistema, Almacenar información, Comunicación del Sistema, Establecer Red, Configuración Módulos de Comunicación y Sensores.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| R 1.0 | Control del sistema | | | | |
| Ref. # | **Función** | **Categoría** | **Atributo** | **Detalle y Limitación** | **Categoría** |
| R.1.1 | Setup | Oculta | Interfaz  Velocidad | El sistema tendrá una respuesta de no más de 10 segundos a solicitudes de configuración | Obligatorio |
| R.1.2 | Loop | Evidente | Interfaz  Velocidad | El sistema será intuitivo para utilizar su ciclo de trabajo asociada a la solicitud de información. | Obligatorio |
| R.1.3 | Confirmar Mensaje | Oculta | Comunicación  Interfaz  Velocidad | El sistema facilitará la confirmación de un mensaje por medio del conteo de la cantidad de datos registrado. | Obligatorio |
| R.1.4 | Control Sensor | Evidente | Comunicación  Interfaz  Velocidad  Específico | El sistema permitirá realizar un control sobre el ciclo de trabajo sobre una petición al sistema | Obligatorio |
| R.1.5 | Memoria Libre | Oculta | Interfaz  Comunicación  Velocidad | El sistema deberá eliminar información de su memoria RAM para mantener un uso correcto de la misma. | Obligatorio |

Tabla 10: Listado de atributos por función del software para el control del sistema.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| R 2.0 | Almacenar información | | | | |
| Ref. # | **Función** | **Categoría** | **Atributo** | **Detalle y Limitación** | **Categoría** |
| R.2.1 | Configurar Memoria | Oculta | Velocidad | El sistema lograra habilitar la memoria para almacenar información en menos de 2 segundos | Obligatoria |
| R.2.2 | Escribir en Memoria | Oculta | Velocidad  Comunicación |  |  |
| R.2.3 | Error en Memoria | Oculta | Velocidad  Interfaz | El sistema podrá responder con alertas ante eventuales fallos en su memoria de almacenamiento | Obligatoria |
| R.2.4 | Leer Memoria | Oculta | Velocidad | El sistema logrará adquirir datos almacenados en memoria en un periodo no mayor a medio segundo. | Obligatoria |
| R.2.5 | Crear | Oculta | Velocidad | El sistema logrará crear espacios de memoria definidos para un conjunto de datos específicos | Obligatoria |
| R.2.6 | Liberar Memoria | Oculta | Velocidad  Rendimiento | El sistema tendrá la capacidad de liberar memoria que no sea de utilidad | Obligatoria |
| R.2.7 | Entregar Datos | Oculta | Velocidad  Interfaz | El sistema podrá entregar datos almacenados en memoria principal o secundaria de manera rápida y fácil | Obligatoria |

Tabla 11: Listado de atributos por función del software para Almacenar Información.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| R 3.0 | Comunicación del Sistema | | | | |
| Ref. # | **Función** | **Categoría** | **Atributo** | **Detalle y Limitación** | **Categoría** |
| R.3.1 | Enviar mensaje | Oculta | Comunicación  Velocidad | El mensaje será enviado entre Nodo Receptor y Nodo Sensor para establecer un flujo de trabajo | Obligatorio |
| R.3.2 | Recibir Mensaje | Oculta | Comunicación  Velocidad | El mensaje será recepcionado y confirmado para su posterior utilización | Obligatorio |
| R.3.3 | Borrar Mensaje | Oculta | Velocidad  Rendimiento | Los mensajes que sean utilizados lograran ser borrados para optimizar el uso de memoria en placas de Arduino | Obligatorio |
| R.3.4 | Convertir Mensaje | Oculta | Velocidad | El sistema podrá convertir un mensaje a numero para un manipulación más expedita | Opcional |
| R.3.5 | Recuperar Mensaje | Evidente | Velocidad  Rendimiento | El sistema podrá recuperar los mensajes almacenados en memoria para su utilización | Obligatorio |
| R.3.6 | Limpiar Buffers | Oculta | Rendimiento  Comunicación  Velocidad | El sistema logrará limpiar los buffer de entrada y salida para no acumular información inútil. | Opcional |
| R.3.7 | Setear Mensaje | Oculta | Rendimiento  Comunicación  Velocidad | El sistema podrá personalizar algún mensaje si lo requiere para propósitos de un flujo de trabajo | Obligatoria |

Tabla 12: Listado de tributos por función del software para la comunicación del sistema.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| R 4.0 | Establecer Red | | | | |
| Ref. # | **Función** | **Categoría** | **Atributo** | **Detalle y Limitación** | **Categoría** |
| R.4.1 | Iniciar Red | Oculta | Rendimiento  Velocidad  Conectividad | El sistema podrá conectarse a un red LAN para comunicaciones por esta. | Obligatorio |
| R.4.2 | Leer mensaje | Oculta | Conectividad  Velocidad | El sistema deberá adquirir los mensajes que sean enviados por la red en menos de 1 segundo | Obligatoria |
| R.4.3 | Eliminar Mensaje | Oculta | Rendimiento  Velocidad | El sistema debe ser capaz de eliminar mensajes que no sean de utilidad | Obligatorio |
| R.4.4 | Procedimiento de Envío | Oculta | Rendimiento  Velocidad  Conectividad | Un procedimiento simplificado para poder lograr la comunicación por medio de una red | Opcional |
| R.4.5 | Envió por Red | Oculta | Conectividad  Velocidad | El sistema enviará por medio de la red a un dirección especifica un mensaje para su procesamiento | Obligatorio |
| R.4.6 | Error de Conexión | Oculta | Rendimiento  Velocidad | El sistema advertirá si algún fallo es percibido en el envió de información por la red | Obligatorio |
| R.4.7 | Enviar Mensaje | Oculta | Conectividad  Velocidad | El sistema realiza la ejecución del envió del mensaje | Obligatorio |

Tabla 13: Listado de atributos por función para establecer red.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| R 5.0 | Configuración Módulos de Comunicación | | | | |
| Ref. # | **Función** | **Categoría** | **Atributo** | **Detalle y Limitación** | **Categoría** |
| R.5.1 | Modo Comando | Oculta | Especifico  Comunicación  Velocidad | El sistema permitirá configurar los módulos de comunicación durante la ejecución de solicitudes | Obligatoria |
| R.5.2 | Configurar parámetros | Oculta | Especifico  Velocidad | El sistema logrará configurar parámetros específicos de los módulos de comunicación | Obligatoria |
| R.5.3 | Salir Modo Comando | Oculta | Velocidad  Rendimiento | El sistema asegurará que las configuraciones sean guardadas correctamente | Obligatoria |
| R.5.4 | Restablecer | Oculta | Rendimiento | El sistema podrá restablecer las configuraciones de los módulos de comunicación. | Opcional |
| R.5.5 | Explorar red | Oculta | Comunicación  Velocidad | El sistema deberá ser capaz de detectar los dispositivos que conforman la red inalámbrica | Opcional |
| R.5.6 | Obtener Equipos | Evidente | Velocidad | El sistema deberá entregar los equipos que conforman la red inalámbrica de sensores | Obligatoria |

Tabla 14: Listado de atributos por función del software para configuración módulos de comunicación.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| R 6.0 | Sensores | | | | |
| Ref. # | **Función** | **Categoría** | **Atributo** | **Detalle y Limitación** | **Categoría** |
| R.6.1 | Transformar Medida | Oculta | Especifico | El sistema permitirá la obtención de un valor métrico desde un valor analógico de sensor | Obligatoria |
| R.6.2 | Obtener Medida | Oculta | Especifico | El sistema logrará obtener un valor analógico procedente de un sensor. | Obligatoria |

Tabla 15 LISTADO DE ATRIBUTOS POR FUNCIÓN DEL SOFTWARE PARA sensores.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| R 7.0 | Servidor | | | | |
| Ref. # | **Función** | **Categoría** | **Atributo** | **Detalle y Limitación** | **Categoría** |
| R.7.1 | Abrir Servidor | Evidente | Interfaz  Velocidad  Conectividad | Al proveer de una dirección ip y un puerto logrará abrir un servidor a la espera de clientes | Obligatoria |
| R.7.2 | Manejar Conexiones | Oculta | Conectividad  Velocidad | Los clientes conectados serán independientes entre sí. | Obligatoria |
| R.7.3 | Función sql | Oculta | Conectividad | La información que provenga desde el cliente será transferida hacia una Base de Dato | Obligatoria |
| R.7.4 | Función parsear | Oculta | Rendimiento | La información que proviene del cliente deberá ser parseada para ejecutar el sql correcto | Obligatoria |

Tabla 16 LISTADO DE ATRIBUTOS POR FUNCIÓN DEL SOFTWARE PARA Servidor.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| R 8.0 | Ventana | | | | |
| Ref. # | **Función** | **Categoría** | **Atributo** | **Detalle y Limitación** | **Categoría** |
| R.8.1 | Listar Puertos | Evidente | Interfaz | El sistema consulta los puertos donde se conectan los dispositivos arduino nodo receptores | Obligatoria |
| R.8.2 | Obtener Equipos | Evidente | Interfaz | Véase 5.6 | Obligatoria |
| R.8.3 | Listar Sensores | Evidente | Interfaz | El sistema listará los sensores pertenecientes a un dispositivos previa consulta a la Base de Datos | Obligatoria |

Tabla 17 LISTADO DE ATRIBUTOS POR FUNCIÓN DEL SOFTWARE PARA Ventana.

Observación: La velocidad está limitada por la transmisión de la comunicación entre los módulos y el protocolo de comunicación utilizado.

## 3.3. Actores

Los principales actores o entes externos que interactúan con el software de describen a continuación.

|  |  |
| --- | --- |
| Actores | |
| Actor | **Descripción** |
| http://martin.elwin.com/blog/wp-content/uploads/2008/05/stick.png  Usuario | El usuario es quien interactuará con el sistema y ejecutará acciones sobre él. Dicho usuario se describe como el encargado de una estación de monitorización |
| http://martin.elwin.com/blog/wp-content/uploads/2008/05/stick.png  BD | Este actor se encargará de almacenar, administrar y mantener toda la información del sistema en una base de datos destinada al software. Dicha base de datos fue desarrollada mediante el gestor de bases de datos MySQL. |
| http://martin.elwin.com/blog/wp-content/uploads/2008/05/stick.png  Arduino | Este actor se encargará de ejecutar las acciones destinadas por el software por medio de sus componentes de hardware. |
| http://martin.elwin.com/blog/wp-content/uploads/2008/05/stick.png  Servidor | Este actor se encargará de recibir peticiones de cliente que envían información correcta para ser almacenada en un Base de Datos. |

## 3.4. Casos de uso

A continuación se presentan los casos de uso establecidos para el funcionamiento del software de acuerdo a los actores que participan en cada uno de ellos, el propósito que poseen, su tipo, el curso normal de eventos y los cursos alternos en caso de que el usuario interactúe de una forma no casual frente al software.

### 3.4.1. Caso de uso esencial

A continuación se describen los casos de casos de uso en formato esencial expandido.

#### Caso 1

|  |  |
| --- | --- |
| ID | C01 |
| Caso de Uso | Detectar Equipos |
| Actores | Usuario, Arduino |
| Propósito | Reconocer los equipos que conformaron la red inalámbrica de los módulos de comunicación |
| Resumen | El usuario ingresará un comando para indicar la acción que debe ejecutar el nodo receptor y de esta manera lograr obtener las direcciones de los equipos que conforman la red. |
| Tipo | Primario y esencial |
| Referencias Cruzadas | R.1.2, R.1.4, R.3.1, R.3.2, R.3.3, R.3.5, R.3.6, R.3.7, R.5.1, R.5.2, R.5.3, R.5.5, R.5.6, R.8.2 |

##### 

##### Curso normal de eventos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
| 1.- El caso de uso inicia cuando un usuario selecciona “Listar Equipos” |  |
|  | 2.-El sistema obtiene la petición del usuario |
|  | 3.-El sistema se conecta con el hardware de arduino. |
|  | 4.-El sistema busca los módulos de comunicación presentes en la red por medio del hardware de arduino. |
|  | 5.-La placa de arduino envía los datos recolectados al sistema. |
|  | 6.- El sistema muestra una lista con las direcciones de cada dispositivo disponible en la red. |
|  | 7.-El sistema se desconecta del hardware. |

##### Cursos Alternos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
|  | 4.-El sistema falla en realizar la búsqueda de dispositivos en la red. Se muestra una notificación de error de scanner. |

#### 

#### Caso 2

|  |  |
| --- | --- |
| ID | C02 |
| Caso de Uso | Solicitar Medidas |
| Actores | Usuario, Arduino |
| Propósito | Solicitar realizar una medida sobre un sensor. |
| Resumen | El usuario puede realizar la petición para obtener medidas de un sensor específico desde un nodo sensor. |
| Tipo | Primario |
| Referencias Cruzadas | R.3.1, R.3.2, R.3.3, R.3.4, R.3.5, R.3.6, R.3.7, R.5.1, R.5.2, R.5.3, R.5.5, R.5.6, R.6.1, R.6.2 |

##### Curso normal de eventos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
| 1.- El caso de uso inicia cuando un usuario selecciona “*Medir*” |  |
|  | 2.-El sistema obtiene la petición del usuario |
|  | 3.-El sistema se conecta con el hardware de arduino. |
|  | 4.- El sistema envía la solicitud para obtener la información desde un nodo sensor por medio del hardware arduino. |
|  | 5.- El hardware arduino recibe la respuesta del nodo sensor para procesar (Véase Caso 3 y 4) y enviar hacia el sistema |
|  | 6.- El sistema recibe la información para ser mostrada en pantalla |
|  | 7.-El sistema se desconecta del hardware. |

##### Cursos Alternos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
|  | 5.- El sistema falla al recibir la información. Solicita nuevamente el dato. |

#### Caso 3

|  |  |
| --- | --- |
| ID | C03 |
| Caso de Uso | Almacenar Datos en micro SD |
| Actores | Arduino, Usuario |
| Propósito | Almacenar información proveniente de sensores |
| Resumen | Arduino deberá almacenará la información en memoria micro SD proveniente de los sensores cuando se solicite. |
| Tipo | Primario |
| Referencias Cruzadas | R.1.2, R.1.2, R.2.1, R.2.2, R.2.3, R.2.6, R.2.7, R.3.5 |

##### Curso normal de eventos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
| 1.- El caso de uso inicia cuando un usuario selecciona “*Medir*” |  |
|  | 2.-El sistema obtiene la petición del usuario |
|  | 3.-El sistema se conecta con el hardware de arduino. |
|  | 4.- El sistema envía la solicitud para obtener la información desde un nodo sensor por medio del hardware arduino. |
|  | 5.- El hardware arduino (nodo receptor) recibe la respuesta del nodo sensor |
|  | 6.-Los datos son almacenados en memoria micro SD.(Véase caso 2 y 4) |
|  | 7.-El sistema se desconecta del hardware. |

##### Cursos Alternos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
|  | 5.- El hardware de Arduino en nodo receptor falla al recibir la información. Solicita nuevamente el dato. |
|  | 6.-El hardware de Arduino en nodo receptor no puede almacenar en micro SD la información. Envía error al sistema e indica error por medio de luces en hardware. |

#### 

#### Caso 4

|  |  |
| --- | --- |
| ID | C04 |
| Caso de Uso | Enviar Datos por Red |
| Actores | Usuario, BD, Arduino, Servidor |
| Propósito | Envió de información para ser guardada en una Base de Datos |
| Resumen | El nodo receptor de Arduino enviará la información recolectada a una Base de datos para su almacenado. |
| Tipo | Primario |
| Referencias Cruzadas | R.1.2, R.3.5, R.4.2, R.4.3, R.4.4, R.4.5, R.4.6, R.4.7 |

##### Curso normal de eventos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
| 1.- El caso de uso inicia cuando un usuario selecciona “*Medir*” |  |
|  | 2.-El sistema obtiene la petición del usuario |
|  | 3.-El sistema se conecta con el hardware de arduino. |
|  | 4.- El sistema envía la solicitud para obtener la información desde un nodo sensor por medio del hardware arduino. |
|  | 5.- El hardware arduino recibe la respuesta del nodo sensor. |
|  | 6.-Los datos son enviados por medio de una red LAN hacia un servidor. (Véase caso 2 y 3) |
|  | 7.-El sistema se desconecta del hardware. |

##### Cursos Alternos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
|  | 5.- El hardware de arduino nodo receptor falla al recibir la información. Solicita nuevamente el dato. |
|  | 6.- El hardware de arduino nodo receptor no puede enviar información por la red. Envía error al sistema e indica error por medio de luces en hardware. |

#### Caso 5

|  |  |
| --- | --- |
| ID | C05 |
| Caso de Uso | Cargar Configuración |
| Actores | Arduino |
| Propósito | Iniciar requerimientos de Hardware de los dispositivos. |
| Resumen | Arduino inicia su configuración para habilitar la utilización de sensores. |
| Tipo | Primario y esencial |
| Referencias Cruzadas | R.1.1, R.4.1 |

##### Curso normal de eventos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
| 1.- El caso de uso inicia cuando el sistema se conecta con el hardware de Arduino |  |
|  | 2.-Arduino inicia configuración de su hardware |

##### Cursos Alternos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
|  | 3.- El sistema falla en conectarse a hardware arduino. |

#### Caso 6

|  |  |
| --- | --- |
| ID | C06 |
| Caso de Uso | Activar Relé |
| Actores | Usuario , Arduino |
| Propósito | Accionar un relé de forma remota de un nodo sensor en particular |
| Resumen | El usuario ordena accionar un relé de un nodo sensor para activar un motor |
| Tipo | Secundario |
| Referencias Cruzadas | R.1.2, R.1.3, R.1.4, R.3.1, R.3.2, R.3.3, R.3.5, R.3.6, R.3.7, R.5.1, R.5.2, R.5.3, R.5.5, R.5.6, |

##### Curso normal de eventos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
| 1.- El caso de uso inicia cuando un usuario selecciona “*Medir*” |  |
|  | 2.-El sistema obtiene la petición del usuario |
|  | 3.-El sistema se conecta con el hardware de arduino. |
|  | 4.- El sistema envía la solicitud para obtener ejecutar la activación del relé por medio del hardware arduino. |
|  | 5.- El hardware arduino recibe la respuesta del nodo sensor. |
|  | 6.- El sistema recibe la información para ser mostrada en pantalla |
|  | 7.-El sistema se desconecta del hardware. |

##### Cursos Alternos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
|  | 3.-El sistema falla en activar el relé. Se muestra una notificación de error. |

#### Caso 7

|  |  |
| --- | --- |
| ID | C07 |
| Caso de Uso | Configuración Módulos |
| Actores | Usuario, Arduino |
| Propósito | Configurar módulos de comunicación |
| Resumen | El usuario enviará peticiones específicas las cuales requiere al módulo configurado con la dirección del nodo sensor requerido. |
| Tipo | Primario y Esencial |
| Referencias Cruzadas | R.1.2, R.5.1, R.5.2, R.5.3 |

##### Curso normal de eventos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
| 1.- El sistema envía solicitud a un nodo sensor. |  |
|  | 2.-El sistema obtiene la petición del usuario |
|  | 3.-El sistema se conecta con el hardware de arduino. |
|  | 4.- El sistema configura el módulo de comunicación. |
|  | 5.- El módulo de comunicación registra la configuración. |
|  | 6.- El módulo de comunicación está habilitado para enviar mensaje al destinatario correcto. |

##### Cursos Alternos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
|  | 2.- El sistema falla al configurar el modulo. Repetir acción hasta lograrlo. |

#### Caso 8

|  |  |
| --- | --- |
| ID | C08 |
| Caso de Uso | Activar Servidor |
| Actores | Usuario, Servidor |
| Propósito | Establecer un Servidor que reciba información |
| Resumen | El usuario establecerá un servidor para que los datos enviados por la red sean procesados por este. |
| Tipo | Primario y Esencial |
| Referencias Cruzadas | R.7.1 |

##### Curso normal de eventos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
| 1.- El caso de uso inicia cuando un usuario ingresa una ip y un puerto para presionar “Encender Servidor” |  |
|  | 2.-El sistema recibe la petición |
|  | 3.-El sistema se conecta el servidor a la red |
|  | 4.-El sistema del servidor queda a la espera de clientes |

##### Cursos Alternos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
|  | 3.-El sistema falla en conectarse a la red. Se muestra una notificación de error de conexión |

#### Caso 9

|  |  |
| --- | --- |
| ID | C09 |
| Caso de Uso | Recibir Conexiones |
| Actores | Arduino, Servidor |
| Propósito | El sistema se encarga de recibir conexiones para el servidor de manera independiente. |
| Resumen | Cuando se solicita una conexión para un cliente, el sistema otorga la conexión. |
| Tipo | Primario |
| Referencias Cruzadas | R.7.2 |

##### Curso normal de eventos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
| 1.- El caso de uso inicia cuando un cliente conecta con el servidor. |  |
|  | 2.-El servidor otorga una conexión al cliente. |
|  | 3.-El servidor recibe la información enviada desde el cliente |
|  | 4.-El servidor procesa la información (Véase caso 10) |
|  | 5.-El cliente se desconecta del servidor |
|  | 6.-El servidor informa que el cliente se desconecto |
|  |  |

##### Cursos Alternos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
|  | 2.-El servidor no logra otorgar una conexión. Indica error de conexión |
|  | 3.-El servidor no recibe correctamente la información. Solicita nuevamente la información |

#### Caso 10

|  |  |
| --- | --- |
| ID | C10 |
| Caso de Uso | Registrar Información |
| Actores | BD, Arduino, Servidor |
| Propósito | Almacenar la información enviada al servidor |
| Resumen | El hardware de Arduino envía la información a un servidor para que este la procese con el fin de almacenar en la base de dato |
| Tipo | Primaria |
| Referencias Cruzadas | R.7.3, R.7.4 |

##### Curso normal de eventos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
| 1.-El caso de uso comienza cuando Arduino nodo receptor (cliente) se conecta con el servidor |  |
|  | 2.- El Servidor obtiene el mensaje |
|  | 3.-El Servidor desglosa el mensaje para crear la inserción correcta en la Base de Datos. |
|  | 4.-El servidor muestra la acción ejecutada |
|  | 5.-El cliente se desconecta del servidor. |
|  | 6.-El servidor informa que el cliente se desconectó. |

##### Cursos Alternos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
|  | 2.-El servidor no logra otorgar una conexión. Indica error de conexión |
|  | 3.-El servidor no recibe correctamente la información. Solicita nuevamente la información |

#### Caso 11

|  |  |
| --- | --- |
| ID | C11 |
| Caso de Uso | Ver puertos |
| Actores | Arduino, Usuario |
| Propósito | Obtener los puertos donde hardware Arduino esté conectado |
| Resumen | Obtener puertos USB donde los arduinos estén conectados para conectar con estos y lograr la interaccion con los nodos sensores. |
| Tipo | Primario y Esencial |
| Referencias Cruzadas | R.8.1 |

##### Curso normal de eventos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
| 1.- El caso de uso inicia cuando el usuario presiona “Ver puertos” |  |
|  | 2.- El sistema consulta los puertos usb utilizados. |
|  | 3.- El sistema entrega una lista de los puertos usb actualmente utilizados. |

##### Cursos Alternos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
|  | 2.- El sistema no encuentra puertos usb. Indica no presencia de dispositivos |

#### Caso 12

|  |  |
| --- | --- |
| ID | C12 |
| Caso de Uso | Listar Sensores |
| Actores | Usuario, BD |
| Propósito | Obtener los sensores que posee un equipo específico. |
| Resumen | El usuario solicita obtener los sensores que posee un nodo sensor en específico por medio de su identificador único. |
| Tipo | Primario |
| Referencias Cruzadas | R.8.3 |

##### Curso normal de eventos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
| 1.- El caso de uso da inicio una vez que un usuario presiona sobre “Seleccionar”. |  |
|  | 2.- El sistema captura la petición. |
|  | 3.-El sistema se conecta a la base de datos propia. |
|  | 4.-El sistema obtiene los sensores del equipo. |
|  | 5.-El sistema se desconecta de la Base de Datos propia. |
|  | 6.-El sistema entrega una lista de sensores pertenecientes al equipo. |

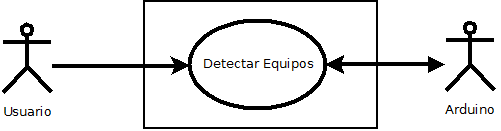
##### Cursos Alternos

|  |  |
| --- | --- |
| Acción del Actor | Respuesta del Sistema |
|  | 3.- El sistema no se conecta a la base de datos. Se muestra una notificación de error de conexión. |
|  | 6.- El sistema no encuentra ningún sensor para el nodo sensor solicitado. Se muestra una lista vacía. |

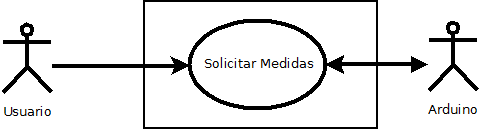
### 3.4.2. Diagrama de Caso de Uso

A continuación se presenta una síntesis gráfica de cada caso de uso.

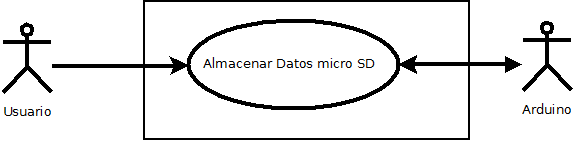
#### C01.- Detectar Equipos



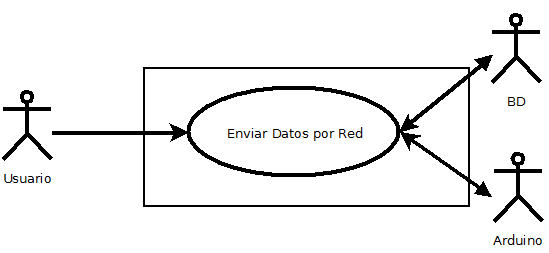
#### C02.- Solicitar Medidas



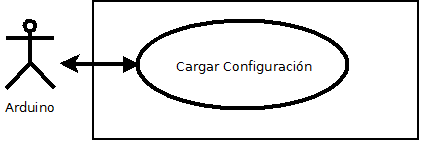
#### C03.- Almacenar Datos Micro SD



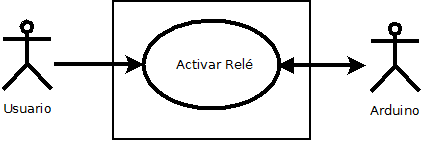
#### C04.- Enviar Datos por Red



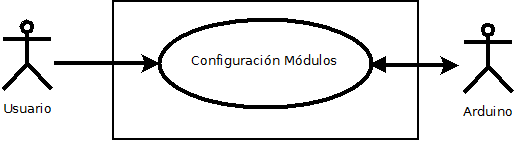
#### C05.- Cargar Configuración



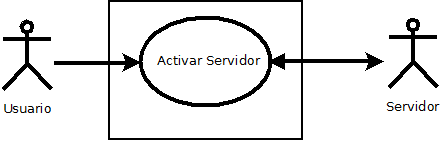
#### C06.- Activar Relé



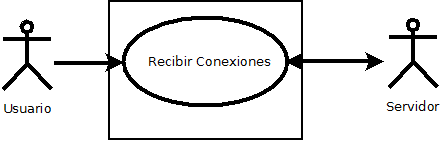
#### C07.- Configurar Módulos



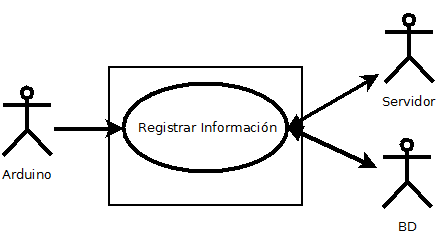
#### C08.- Activar Servidor



#### C09.- Recibir Conexiones



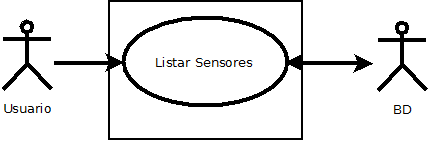
#### C10.- Registrar Información



#### C11.- Ver Puertos



#### C12.- Listar Sensores



A continuación se presenta la relación entre todos los actores y todos los casos de uso.

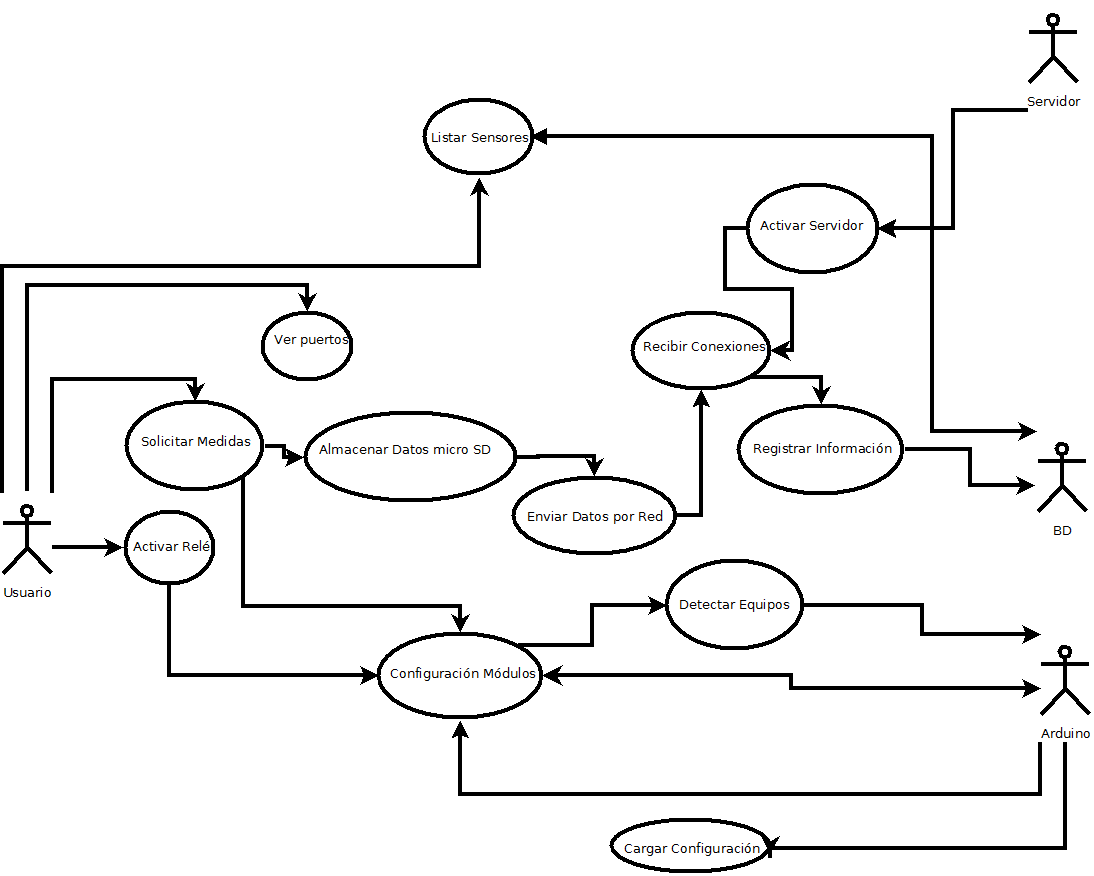


Figura 2: Esquema general de actores y casos de uso.

### 3.4.3. Modelo Conceptual

A continuación se realiza una descripción gráfica de conceptos, asociaciones y atributos del software.

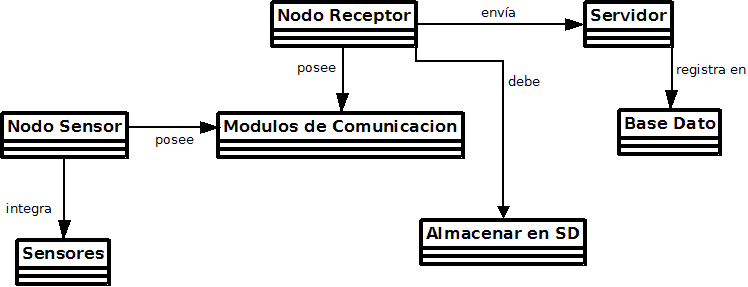


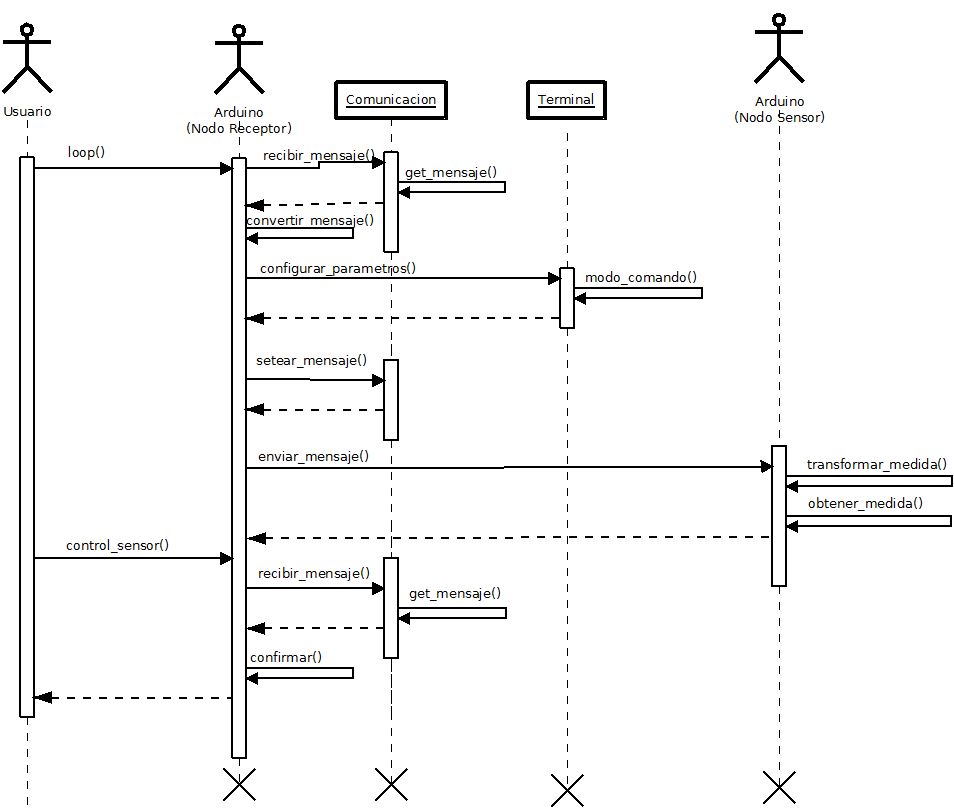
Figura 3: Modelo conceptual del software.

### 3.4.4. Diagrama de Secuencia o Colaboración

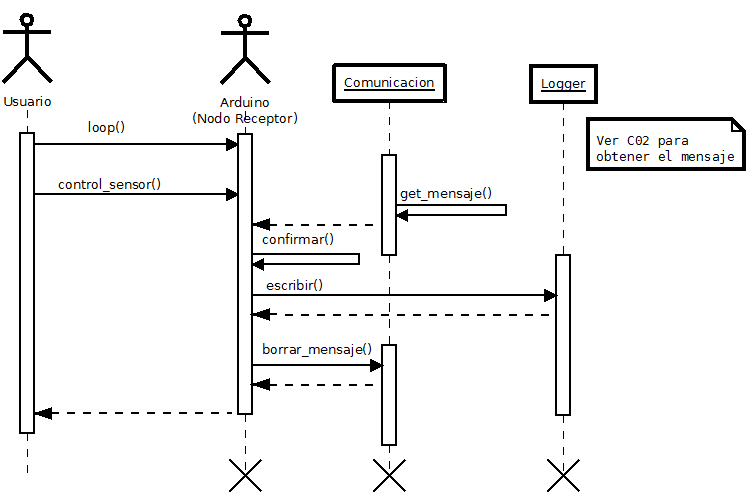
A continuación se presenta una descripción gráfica de eventos entre actores y el sistema. Cabe señalar que se hará referencia a Arduino Nodo Receptor y Arduino Nodo Sensor para indicar la interacción entre estos equipos.

#### C01.- Detectar Equipos C:\Users\Gonzalo\Desktop\Escritorio\Informe\4 Diagramas de Secuencia\Detectar_equipos.png

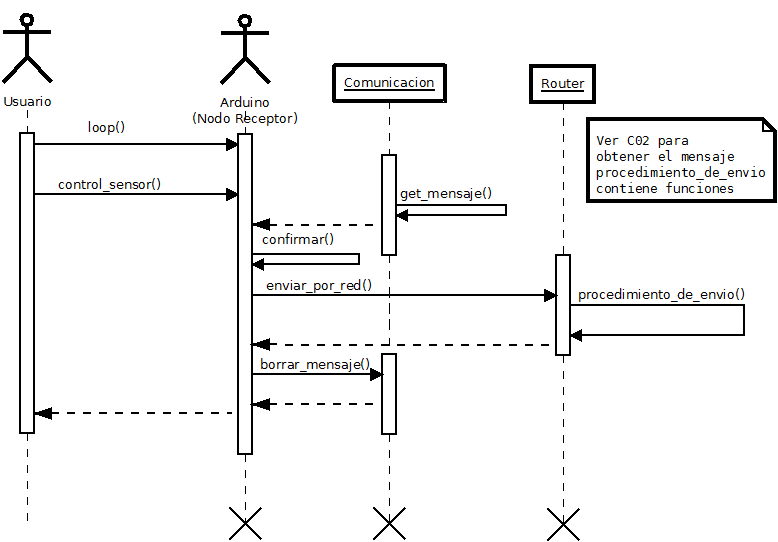
#### C02.- Solicitar Medidas



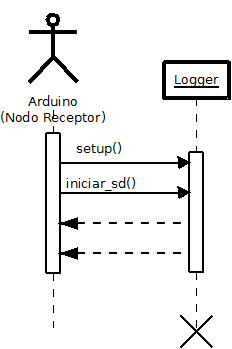
#### C03.-Almacenar Datos Micro SD



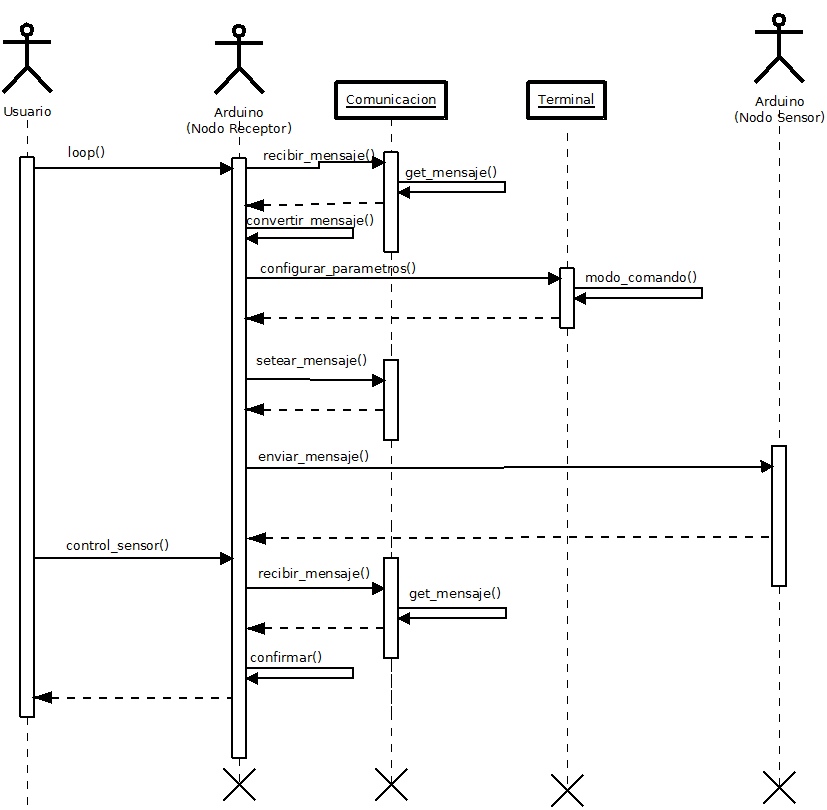
#### C04.- Enviar Datos por Red



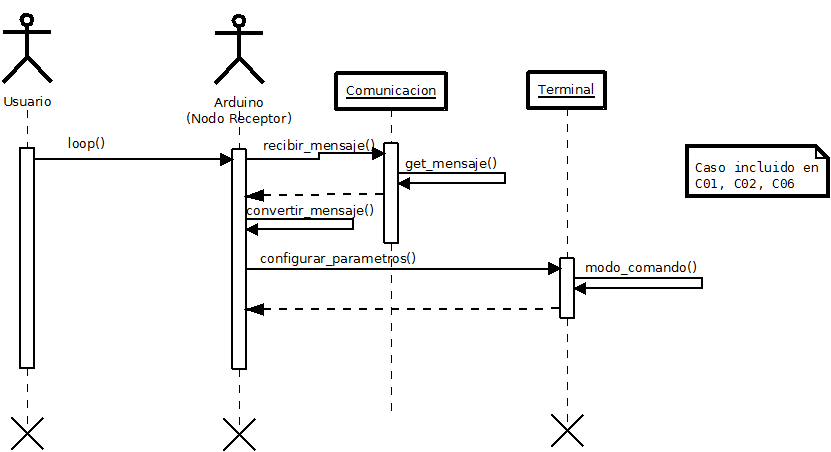
#### C05.- Cargar Configuración



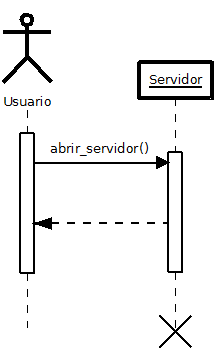
#### C06.- Activar Relé



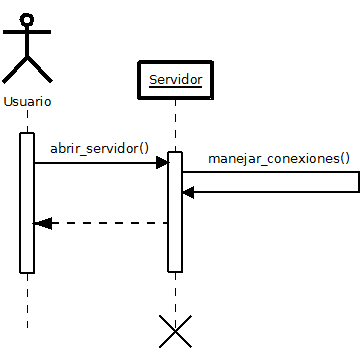
#### C07.- Configuración Módulos



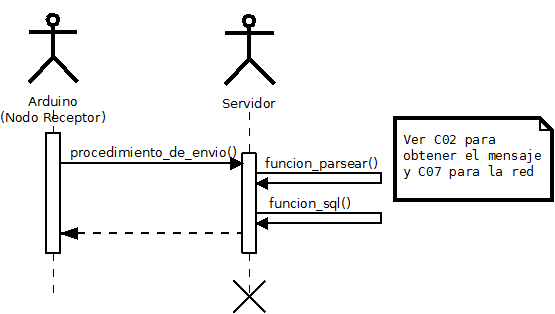
#### C08.- Activar Servidor



#### C09.- Recibir Conexiones

****

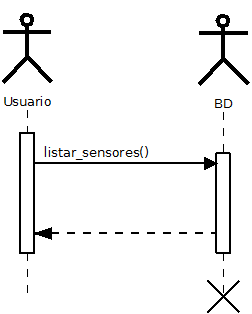
#### C10.- Registrar Información



#### C11.- Ver Puertos

Para el caso del diagrama de interaccion de C11, se interactúa con el Sistema Operativo que aloja la aplicación para realizar la pregunta de los dispositivos usb conectados actualmente. Dicha pregunta es realizada por el software de la aplicación gráfica. Al obtener los dispositivos se entrega una lista.

#### C12.- Listar Sensores



### 3.4.5. Priorización

A continuación se presenta una tabla con las categorías de priorización de los casos de uso según su puntaje.

|  |  |
| --- | --- |
| Clasificación | Puntaje |
| Alto (Esencial) | 21-30 Puntos |
| Medio (Importante) | 12-20 Puntos |
| Bajo (Desechable) | 1-11 Puntos |

Tabla 18: Tabla de puntajes de priorización, se describen tres categorías Alto, Medio y Bajo según su puntaje.

Las cualidades para calificar a los casos de uso se presentan en la siguiente tabla, en la cual, mientras más alto sea el puntaje, más compleja se torna una cualidad.

|  |  |  |
| --- | --- | --- |
| Código | Cualidad | Puntaje |
| A | Tiene una fuerte repercusión en el diseño arquitectónico; incorpora muchas clases a la capa del dominio o requiere servicios de persistencia. | [1 – 5] |
| B | Con relativamente poco esfuerzo obtiene información e ideas importantes sobre el diseño. | [1 – 5] |
| C | Incluye funciones riesgosas, urgentes o complejas. | [1 – 5] |
| D | Requiere una investigación a fondo o tecnología nueva o riesgosa. | [1 – 5] |
| E | Representa los procesos primarios de consultas a base de datos. | [1 – 5] |
| F | Representa los procesos primarios de interacción del usuario con imágenes. | [1 – 5] |

Tabla 19: Tabla de cualidades para la clasificación de los casos de uso.

Con los datos de las tablas anteriores, la priorización de los casos de uso queda de la siguiente forma.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Caso de Uso | A | B | C | D | E | F | Puntaje | Calificación |
| 1 | 5 | 4 | 4 | 5 | 0 | 4 | 22 | Alto |
| 2 | 5 | 5 | 5 | 5 | 0 | 5 | 25 | Alto |
| 3 | 4 | 2 | 5 | 4 | 0 | 3 | 18 | Medio |
| 4 | 4 | 3 | 5 | 5 | 0 | 4 | 21 | Alto |
| 5 | 5 | 5 | 5 | 5 | 0 | 0 | 20 | Medio |
| 6 | 3 | 5 | 5 | 5 | 0 | 0 | 18 | Medio |
| 7 | 5 | 1 | 5 | 5 | 0 | 0 | 16 | Medio |
| 8 | 5 | 1 | 5 | 3 | 5 | 5 | 24 | Alto |
| 9 | 5 | 1 | 5 | 3 | 5 | 5 | 24 | Alto |
| 10 | 5 | 3 | 5 | 2 | 5 | 3 | 23 | Alto |
| 11 | 2 | 2 | 2 | 2 | 0 | 3 | 11 | Bajo |
| 12 | 5 | 1 | 5 | 3 | 5 | 5 | 24 | Alto |

Tabla 20: Tabla de priorización de los casos de uso del software WSN v1.0.

## 3.5. Modelo de dominio

A continuación se describe el modelo de dominio del software, en dicha instancia se describen las entidades, un esquema del modelo y la matriz de rastreabilidad asociada a este.

### 3.5.1. Entidades Reconocidas

En la siguiente sección, se listan y describen las entidades reconocidas para el presente software. Estas fueron identificadas según su propia definición (Véase Glosario Sección 7), además de reconocer aquellas que aparecían en más de un caso de uso.

**Entidad 1.-Nodo:** Equipos registrados en el sistema que poseen un perfil específico. Ellos tienen la clasificación de Nodos Sensores o Nodos Recolectores. Los nodos sensores poseen los sensores que registraran datos. Estos equipos poseen la capacidad de adquirir datos desde el ambiente y enviarlos por medio de una señal inalámbrica a un Nodo Recolector. El Nodo Recolector se encarga de enviar la información para ser almacenada.

**Entidad 2.- Sensores:** Dispositivos que perciben el ambiente y entregan un valor determinado por la influencia del ambiente sobre él. Ellos se encuentran conectados físicamente a determinados Nodos Sensores.

**Entidad 3.- Ubicación:** Información de relevancia para lograr determinar la ubicación geográfica de los nodos sensores que pueden estar en diferentes locaciones.

**Entidad 4.- Historial:** Listado de información la cual contiene los datos registrados por lo nodos sensores repartidos en diferentes locaciones y con diferentes sensores.

**Entidad 5.- Sensores en Nodos:** Entidad que permite reconocer que nodos sensores poseen conectados sensores. Para poder obtener una mejor claridad de las relaciones.

**Entidad 6.- Base de Datos:** Entidad que almacena los datos registrados correspondientes al software.

### 3.5.2. Modelo de Dominio

A continuación se presenta el modelo de entidades agrupadas en paquetes de unidad conceptual junto a sus dependencias.

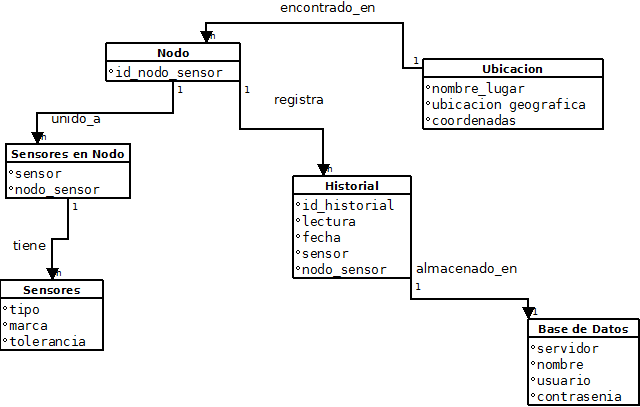


Figura 4: Modelo de dominio, entidades agrupadas en paquetes de unidad conceptual junto a sus dependencias.

### 3.5.3. Matriz de Rastreabilidad

En la siguiente tabla se identifican las entidades reconocidas en el software.

|  |  |
| --- | --- |
| ID.ER # | Entidades |
| 1 | Nodo Sensor |
| 2 | Sensores |
| 3 | Ubicación |
| 4 | Historial |
| 5 | Sensores en nodos |
| 6 | Base de Datos |

Tabla 21: Identificación de las entidades del software.

A continuación se identifican los casos de uso descritos en la sección 3.4.

|  |  |
| --- | --- |
| ID.CU # | Caso de uso |
| C01 | Detectar Equipos |
| C02 | Solicitar Medidas |
| C03 | Almacenar Datos en micro sd |
| C04 | Enviar Datos por Red |
| C05 | Cargar Configuración |
| C06 | Activar Relé |
| C07 | Configuración Módulo |
| C08 | Activar Servidor |
| C09 | Recibir Conexiones |
| C10 | Registrar Información |
| C11 | Ver puertos |
| C12 | Listar Sensores |

Tabla 22: Identificación de los casos de uso del software.

Dada las tablas anteriores, se presenta una matriz donde se relacionan las entidades identificadas con los casos de uso del software.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| CU/ER | 1 | 2 | 3 | 4 | 5 | 6 |
| C01 | X |  | X |  |  |  |
| C02 | X | X | X | X | X | X |
| C03 | X | X |  |  | X |  |
| C04 | X |  |  | X |  | X |
| C05 | X |  |  |  |  |  |
| C06 | X | X |  |  | X |  |
| C07 | X |  | X |  |  |  |
| C08 | X |  |  |  |  |  |
| C09 | X |  |  |  |  |  |
| C10 | X | X | X | X | X | X |
| C11 | X |  |  |  |  |  |
| C12 | X | X |  |  | X |  |

Tabla 23: Matriz de rastreabilidad del software. Se identifica la relación entre las entidades reconocidas y los casos de uso a los que el software se verá enfrentado.

# 4. Validación

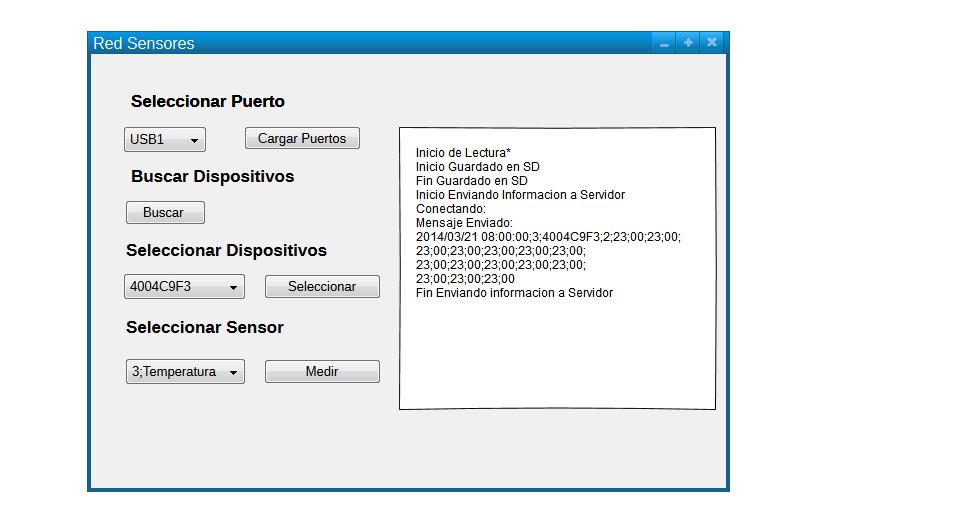
En esta sección se presentan los prototipos del sistema, su utilización y descripción según los casos de uso descritos en la sección 3.4.

## 4.1. Prototipo de validación funcional

A continuación se presentan los prototipos del sistema correspondientes los casos de uso de alta prioridad.

**C01.- Detectar Equipos**

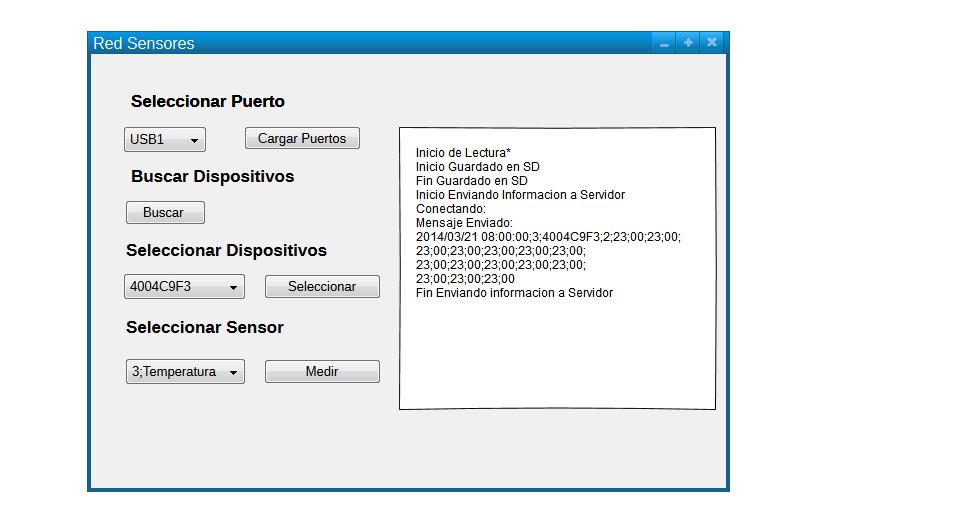
* **Página inicial del usuario.**

****

Dispositivos encontrados en la red sensorial.

“Buscar” Lista los dispositivos que se encuentran en la red.

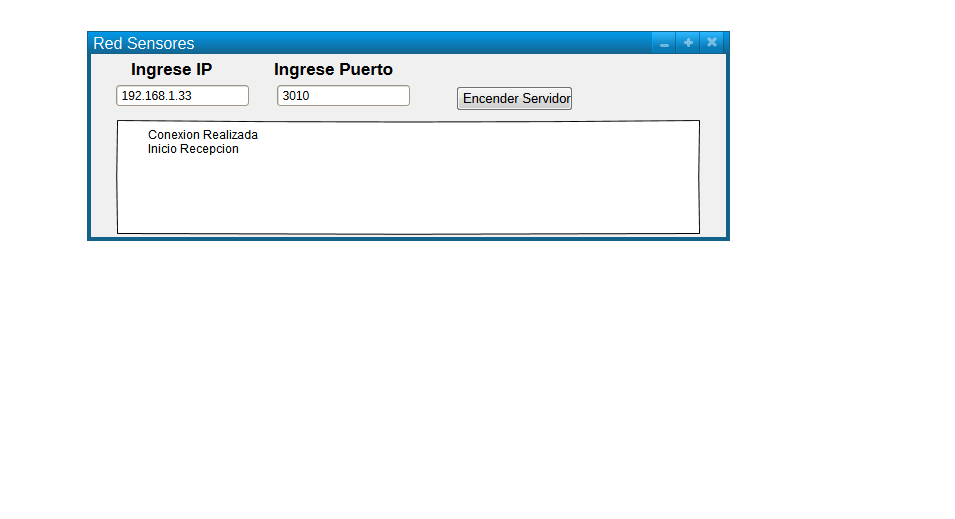
**C02 Solicitar Medidas**

****

Registro que puede ver el usuario al finalizar la orden de Medir

“Medir” Ordena al dispositivo nodo sensor que censé en el ambiente y entregue los valores registrados.

**C08 Activar Servidor**

****

“Encender Servidor” utiliza la ip y el puerto ingresado para encender el servidor con esos datos.

Indicación de la activación

Botón para formular el documento.

Secciones del syllabus, en estas el profesor deberá ingresar toda la información del nuevo syllabus. Dichas secciones forman un estándar único para la formulación de los syllabus el cual contiene:

* Unidad responsable de la construcción del syllabus.
* Contribución a la formación.
* Subcompetencias a desarrollar.
* Unidades de aprendizaje.
* Metodología a utilizar.
* Evaluaciones.
* Requerimientos especiales.
* Bibliografía.

# 5. Diseño

En esta sección se propone y se describe la solución propuesta, relacionándola con lo solicitado.

## 5.1. Derivación del Modelo de Software

A continuación se presenta un modelo de software inicial diagrama de clases y diagramas de estado relacionados a dicho modelo.

### 5.1.1. Modelo de software inicial

En esta sección se presenta la identificación de clases del modelo de dominio las cuales se usan como base del software.

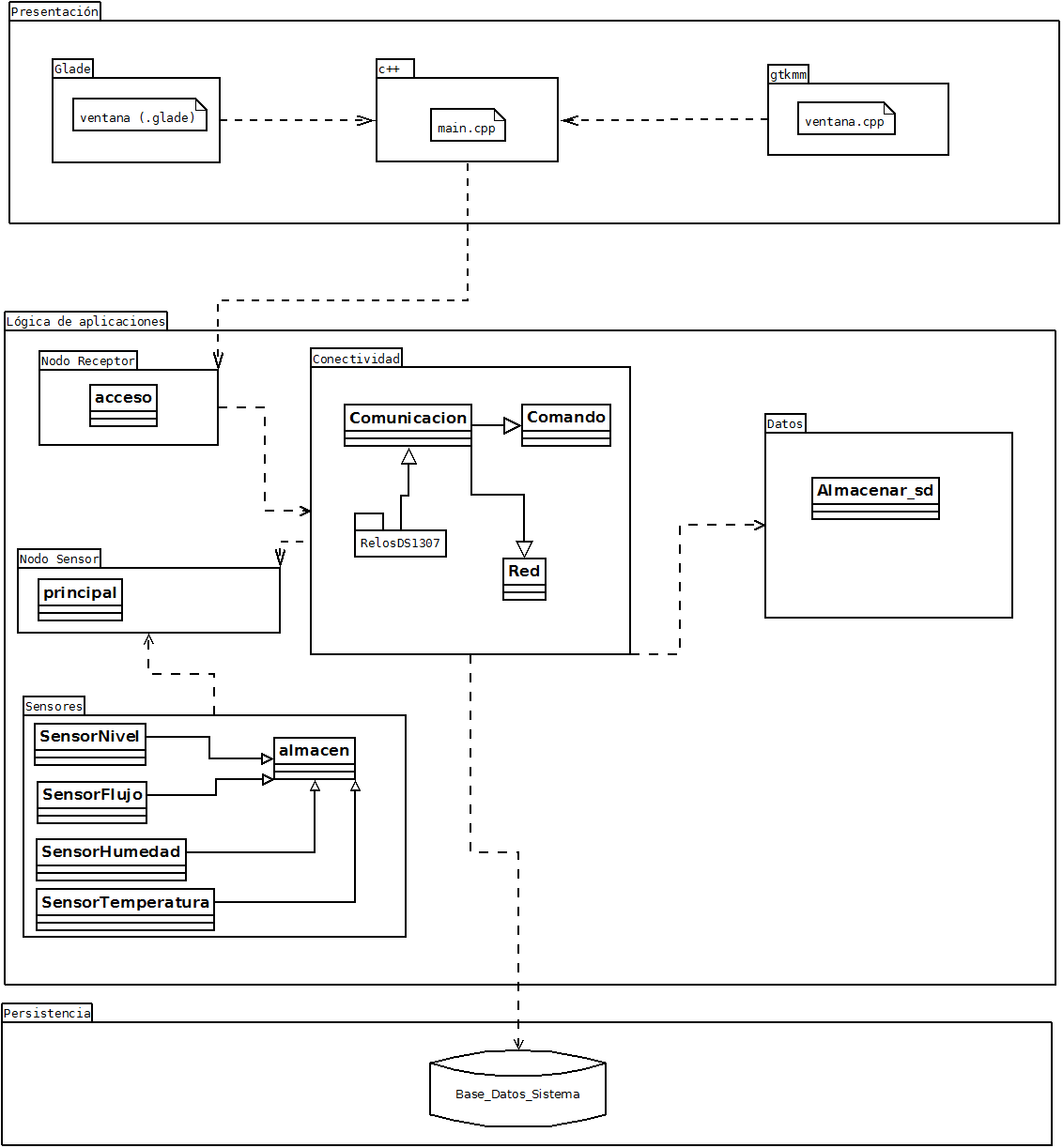


Figura 5: Modelo de software inicial del sistema correspondiente al modelo de tres capas, se presentan los paquetes y clases las cuales se usan como base del software.

### 5.1.3. Diagrama de Clases

A continuación se presenta el diagrama de clases de diseño para el software. En este caso al utilizar hardware distinto se prepara el diagrama de clases para cada nodo, primero veremos el diagrama de clases de nodo receptor y luego el diagrama de clases para el nodo sensor.

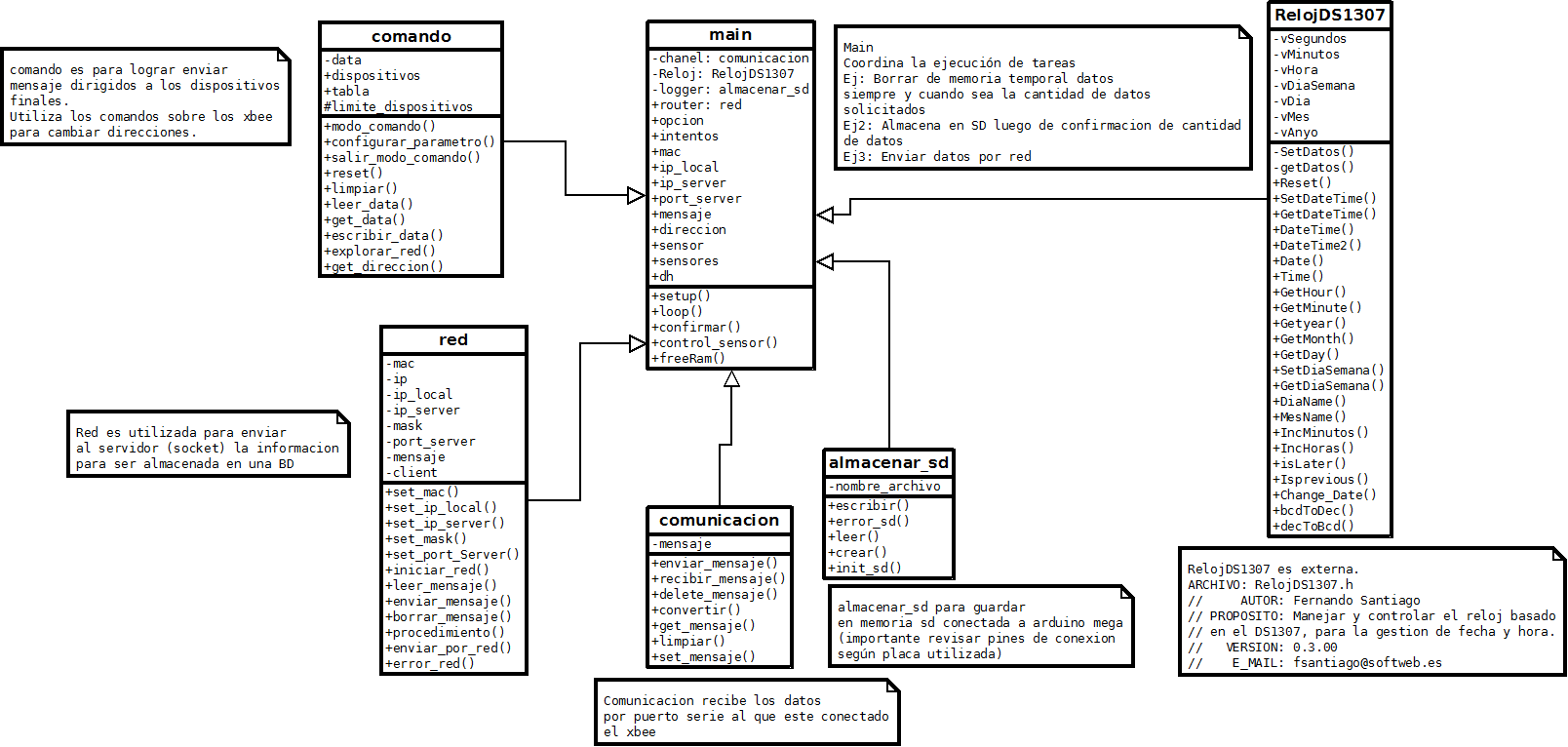


Figura 6: Diagrama de clases de diseño del software, junto con sus atributos y sus respectivos métodos para hardware nodo receptor.

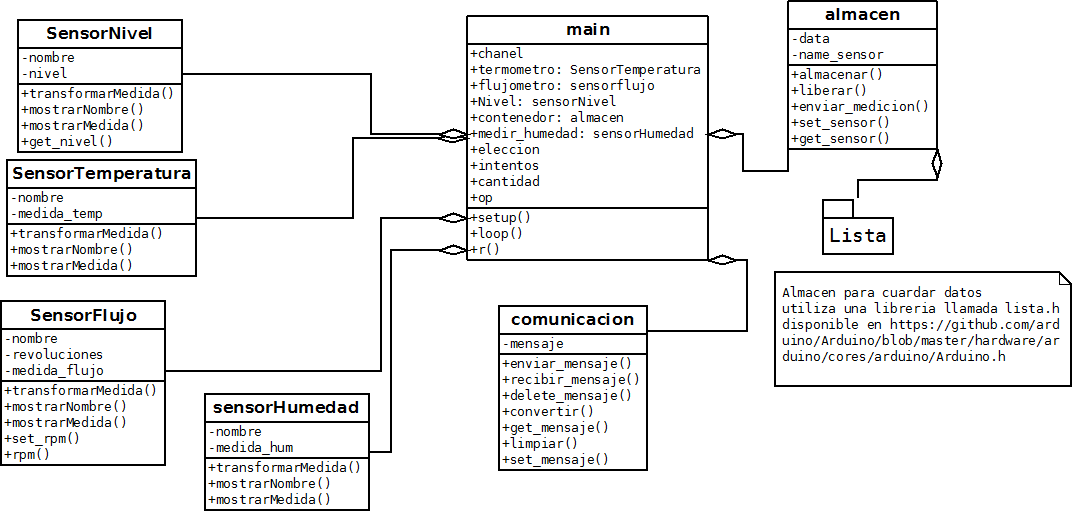
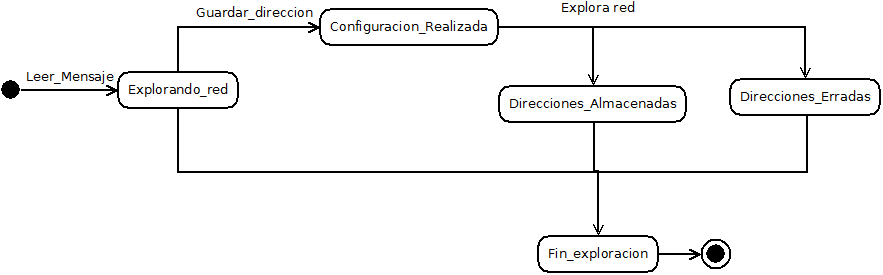


Figura 7 Diagrama de clases de diseño del software, junto con sus atributos y sus respectivos métodos para hardware nodo Sensor.

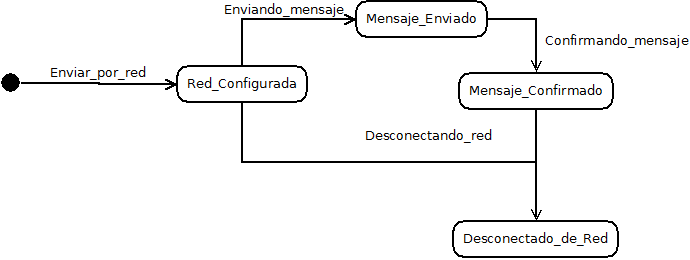
### 5.1.4. Diagramas de Estados

A continuación se presentan los diagramas de estado para las clases reconocidas anteriormente. Primero para hardware nodo receptor y luego nodo sensor

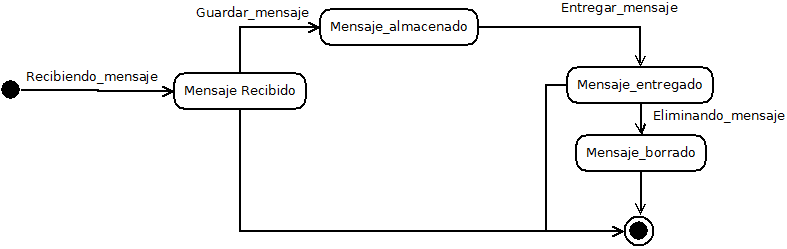
**Clase.-Comando**

****

**Clase.-Red**

****

**Clase.-Comunicación**

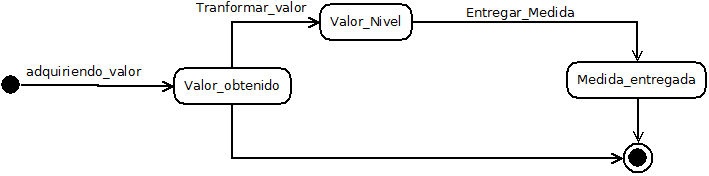
****

**Clase.-Almacenar SD**

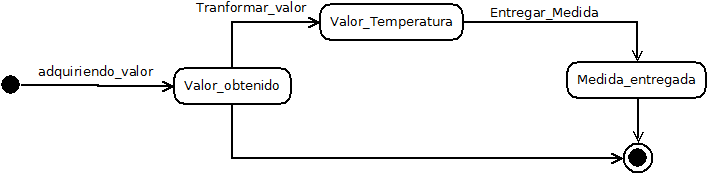
**C:\Users\Gonzalo\Desktop\Escritorio\Informe\2 Diagramas\Diagrama de Estado\almacenar_sd.png**

Para hardware nodo sensor

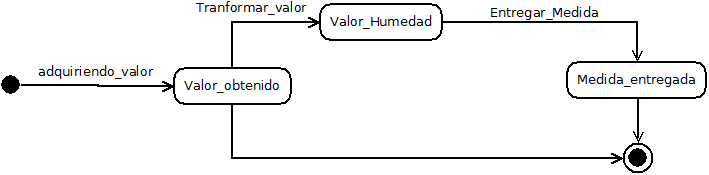
**Clase.-Sensor Nivel**

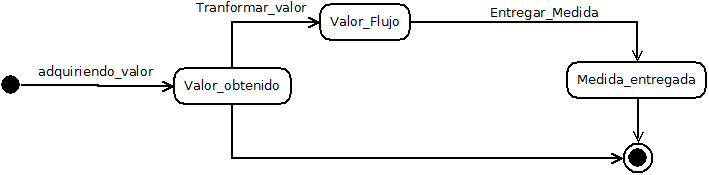
****

**Clase.-Sensor Temperatura**

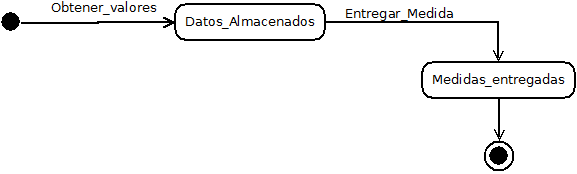
****

**Clase.-Sensor Humedad**

****

**Clase.-Sensor Flujo**

**Clase.-Almacén**



### 5.1.5 Patrones GRASP

Aplicación de los siguientes patrones a las clases presente en modelo de clases para nodo receptor:

|  |  |
| --- | --- |
| Clase | Comando |
| Patrón | Bajo acoplamiento |
| Responsabilidad | La creación de esta clase, surge debido la necesidad de interactuar con los módulos de comunicación de manera única y coordinada cuando se requieran de la transmisión de mensajes particulares a equipos remotos. Se buscaba evitar que otras clases tuvieran acceso a los módulos de comunicación, para tener el recurso controlado únicamente por esta clase para evitar acoplamiento. Por tanto con la aplicación del patrón, fue posible visualizar cómo esta clase se hacía estrictamente necesaria para reducir el acoplamiento dentro del sistema. |

|  |  |
| --- | --- |
| Clase | Main |
| Patrón | Controlador |
| Responsabilidad | La característica de esa clase es la posibilidad de reutilizar los objetos creados para interactuar con los requerimientos presentes del usuario. Para lograr actuar de modo intermediario entre las clases presentes y lograr obtener un flujo de trabajo. |

|  |  |
| --- | --- |
| Clase | Red |
| Patrón | Bajo Acoplamiento |
| Responsabilidad | A partir del objeto creado con la clase red se genera una sola conexión por equipo para interactuar con el servidor. De esta forma se mantiene concentrada en una clase la capacidad de enviar información para ser almacenada sin necesidad de utilizarla la conectividad en otra clase. Es por ello la importancia en la existencia de la presente clase. |

|  |  |
| --- | --- |
| Clase | Comunicación |
| Patrón | Alta Cohesión |
| Responsabilidad | Con el fin de obtener un mensaje claro y sin errores entre la comunicación inalámbrica de equipos requiere de la capacidad de la clase de comunicación para tener una alta cohesión de procedimiento. |

# 6. Implantación

Esta sección pretende mostrar a los instaladores y administradores del sistema las partes a entregar y su interacción.

## 6.1. Código fuente completo (parcial)

A continuación se presenta el código fuente del sistema desarrollado.

### 6.1.1 Clase Comando

#ifndef comando\_h

#define comando\_h

#include "Arduino.h"

/\*LIMITE PARA REGISTRAR DISPOSITIVOS\*/

#define limite\_dispositivos 5

class comando{

public:

comando();

~comando();

void modo\_comando();

/\*void leer\_configuracion();\*/

void configurar\_parametro(String DL,String DH);

void salir\_modo\_comando();

void reset();

void limpiar();

void leer\_data();

String get\_data();

void escribir\_data();

void explorar\_red();

String get\_direccion(int a);

private:

String data;

int dispositivos;

String tabla[limite\_dispositivos];

};

#endif

### 6.1.2. Clase Red

/\*

\* File: red.h

\* Author: gonzalo

\*

\*/

#ifndef RED\_H

#define RED\_H

#include <SPI.h>

#include <Ethernet.h>

class red{

public:

red();

~red();

void set\_mac(byte\* mac);

void set\_ip\_local(byte\* ip);

void set\_ip\_server(byte\* ip);

void set\_mask(byte\* mask);

void set\_port\_server(int port);

int iniciar\_red();

int leer\_mensaje();

void enviar\_mensaje();

void borrar\_mensaje();

String get\_mensaje();

void set\_mensaje(String msg);

int server\_disponible();

int procedimiento(String msg);

void enviar\_por\_red(String data);

void error\_red(int op);

private:

byte \*mac;

byte \*ip;

byte \*ip\_local;

byte \*ip\_server;

byte \*mask;

int port\_server;

String mensaje;

EthernetClient client;

};

#endif /\* RED\_H \*/

### 6.1.3. Clase Comunicación

#ifndef comunicacion\_h

#define comunicacion\_h

#include "Arduino.h"

class comunicacion{

public:

comunicacion();

~comunicacion();

int enviar\_mensaje(int uart);

int recibir\_mensaje(int uart);

void delete\_mensaje();

int convertir(String numero);

String get\_mensaje();

void limpiar();

void set\_mensaje(String msg);

private:

String mensaje;

};

#endif

### 6.1.4 Clase Almacenar SD

#ifndef almacenar\_sd\_h

#define almacenar\_sd\_h

#include "Arduino.h"

class almacenar\_sd{

public:

almacenar\_sd();

~almacenar\_sd();

int escribir(String data,String nombre);

void error\_sd(int op);

/\*Funciones Futuras

void leer();

int crear(String name);

\*/

void init\_sd();

};

#endif

### 6.1.5. Clase Sensor Nivel

#ifndef sensorNivel\_h

#define sensorNivel\_h

class sensorNivel {

public:

sensorNivel();

double transformarMedida();

double mostrarMedida();

char \* mostrarNombre();

double get\_nivel();

private:

double nivel;

char\* \_nombre;

};

#endif

### 6.1.6. Clase Sensor Temperatura

#ifndef sensorHumedad\_h

#define sensorHumedad\_h

class sensorHumedad {

public:

sensorHumedad();

double transformarMedida();

double mostrarMedida();

char \* mostrarNombre();

private:

char\* \_nombre;

double medida\_hum;

};

#endif

### 6.1.7. Clase Sensor Humedad

#ifndef sensorTemperatura\_h

#define sensorTemperatura\_h

class sensorTemperatura {

public:

sensorTemperatura();

double transformarMedida();

double mostrarMedida();

char \* mostrarNombre();

private:

char\* \_nombre;

double medida\_temp;

};

#endif

### 6.1.8. Clase Sensor Flujo

#ifndef sensorflujo\_h

#define sensorflujo\_h

class sensorflujo {

public:

sensorflujo();

double transformarMedida();

double mostrarMedida();

char \* mostrarNombre();

void rpm ();

void set\_rpm();

public:

double flujo;

char\* \_nombre;

double revoluciones;

};

#endif

### 6.1.9. Clase Almacen

#ifndef almacen\_h

#define almacen\_h

#include "Lista.h"

#define alerta\_insercion 3

class almacen{

public:

almacen();

~almacen();

void almacenar(double value);

void liberar();

String enviar\_medicion();

void set\_sensor(String nombre);

String get\_sensor();

private:

LList<double> data;

String name\_sensor;

};

#endif

## 6.2. Dependencias

A continuación se indican qué productos, componentes o bibliotecas son requeridos para que el sistema pueda operar normalmente.

#### 6.2.1. Conexión a Internet

Algo crucial en el correcto funcionamiento del software es una conexión estable a Internet, ya que la interacción cliente-servidor es mediante el protocolo TCP (puerto 3010).

El Software que se debe cargar en hardware nodo receptor (placa arduino mega 2560) debe contemplar la edición de la ip para el servidor y la ip del cliente para su funcionamiento, además de cambiar el puerto de conexión.

#### 6.2.2. Paquete mysql, g++, gtkmm 3.0,

La obtención de la información desde la base de datos del sistema y las clases implementadas en este fueron desarrolladas en C++ por lo que se requiere que esté instalado g++ y la librería mysql.h en la máquina para tenga un correcto funcionar. La aplicación utiliza gtkmm3.0 para su despliegue.

#### 6.2.3. Sistema de gestión de bases de datos MySQL

El almacenamiento de la información de todo el sistema fue desarrollado en torno a un modelo relacional de base de datos con la ayuda del sistema de gestión de base datos MySQL, por lo que al igual que en el caso anterior, dicho recurso debe estar instalado en la máquina que actuará como servidor.

#### 6.2.4. Servidor apache

Para la correcta conexión de los usuarios al software mediante Internet es necesario tener instalada una versión de apache estable y funcional en la máquina que funcionará como servidor.

# 7. Anexos

## 7.1. Glosario

Actor: En el Lenguaje Unificado de Modelado (UML), un actor "especifica un rol jugado por un usuario o cualquier otro sistema que interactúa con el sujeto. (OMG Unified Modeling Language [OMG UML]).

Caso de uso: Un caso de uso es una descripción de los pasos o las actividades que deberán realizarse para llevar a cabo algún proceso. (Wikipedia).

Componente de software: Un componente de software es un elemento de un sistema software que ofrece un conjunto de servicios, o funcionalidades, a través de interfaces definidas. (Wikipedia).

Diagrama de interacción: El diagrama de interacción, representa la forma en como un Cliente (Actor) u Objetos (Clases) se comunican entre si en petición a un evento. Esto implica recorrer toda la secuencia de llamadas, de donde se obtienen las responsabilidades claramente. (Patricio Salinas, Universidad de Chile).

Diagrama de secuencia: El diagrama de secuencia es un tipo de diagrama usado para modelar interacción entre objetos en un sistema según UML. (Wikipedia).

Modelo de dominio: El modelo de dominio muestra (a los modeladores) clases conceptuales significativas en un dominio del problema; es el artefacto más importante que se crea durante el análisis orientado a objetos. (C. Larman. 2003).

Matriz de rastreabilidad: Es un recurso que los desarrolladores crean para controlar la evolución y cambios en los requisitos; dependerá de los modelos de desarrollo (requisitos, entidades, casos de uso, clases, etc.) que hayan sido construidos por los desarrolladores. (M. Silva, et al. 2007).

MySQL: Es el sistema de gestión de base de datos de código abierto más popular del mundo. (MySQL.com).

Patrones GRASP: Los patrones GRASP describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en formas de patrones. GRASP es un acrónimo que significa General Responsibility Assignment Software Patterns. El nombre se eligió para indicar la importancia de captar estos principios, si se quiere diseñar eficazmente el software orientado a objetos. (http://www.practicasdesoftware.com.ar/).

Programación en capas: En la programación en capas los objetos se dividen según su funcionalidad. Destacan tres principales: la Capa de Interfaz o Frontera, compuesta por los objetos encargados de interactuar con el usuario, como lo son los formularios e interfaces de la aplicación; por otra parte está la Capa de Lógica de Negocio o Control, en donde se encuentran los objetos que realizan la mayor parte del trabajo interno del programa, en esta etapa destaca la lógica de la aplicación así como la funcionalidad de servir de enlace entre las otras capas; por último se encuentra la Capa de Datos, integrada por los objetos que envían y obtienen información al comunicarse con bases de datos u otros sistemas de información que colaboran con el programa.

UML: Lenguaje Unificado de Modelado (LUM o UML, por sus siglas en inglés, Unified Modeling Language) es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad; está respaldado por el OMG (Object Management Group). Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. (Wikipedia).

# 8. Bibliografía

1.- Desarrollo e implementación utilizando Arduino y Zigbee con un sensor ultrasónico para control de nivel de llenado <http://deeea.urv.cat/public/PROPOSTES/pub/pdf/2114pub.pdf>

2.- Desarrollo de sistema inmótico basado en plataforma Arduino <http://unicarlos.com/_ARDUINO/Rfid_WK200/Proyecto_3_Memoria_UniversidadLaboral.pdf>

3.- Diseño de una plataforma modular para el desarrollo agil de aplicaciones basado en modulos xbee programables <http://biblioteca.unirioja.es/tfe_e/R000001350.pdf>

4.- Faludi, R.; Building Wireless Sensor Networks; O'Reilly: Cambridge, 2011

5.- <http://www.olimex.cl/pdf/Wireless/ZigBee/XBee-Guia_Usuario.pdf>

6.- <http://www.flytron.com/>

7.- <http://arduino.cc/>