

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
1.1. PROPÓSITO	3
1.2. DESCRIPCIÓN BREVE DEL PROBLEMA	3
2.1. PLANIFICACIÓN TEMPORAL DE ACTIVIDADES	4
2.1.1. INFORME GENERAL	4
3. ANÁLISIS	4
3.1. CONTEXTO	4
3.1.1. DESCRIPCIÓN GENERAL	4
3.1.2. DESCRIPCIÓN DE CLIENTES Y USUARIOS	5
3.2. ESPECIFICACIÓN DE REQUERIMIENTOS	5
3.2.1. FUNCIONES DEL SISTEMA	5
3.2.2. ATRIBUTOS DEL SISTEMA	9
3.2.3. ATRIBUTOS POR FUNCIÓN	10
3.3. ACTORES	16
3.4. CASOS DE USO	17
3.4.1. CASO DE USO ESENCIAL	17
3.4.2. DIAGRAMA DE CASO DE USO	28
3.4.3. MODELO CONCEPTUAL	32
3.4.4. DIAGRAMA DE SECUENCIA O COLABORACIÓN	32
3.4.5. PRIORIZACIÓN	38
3.5. MODELO DE DOMINIO	40
3.5.1. ENTIDADES RECONOCIDAS	40
3.5.2. MODELO DE DOMINIO	41
3.5.3. MATRIZ DE RASTREABILIDAD	41
4. VALIDACIÓN	44
4.1. PROTOTIPO DE VALIDACIÓN FUNCIONAL	44
5. DISEÑO	46
5.1. DERIVACIÓN DEL MODELO DE SOFTWARE	46
5.1.1. MODELO DE SOFTWARE INICIAL	46
5.1.3. DIAGRAMA DE CLASES	47
5.1.4. DIAGRAMAS DE ESTADOS	49
5.1.5. PATRONES GRASP	51
6. IMPLANTACIÓN	52
6.1. CÓDIGO FUENTE COMPLETO (PARCIAL)	52
6.1.1. CLASE COMANDO	52
6.1.2. CLASE RED	53
6.1.3. CLASE COMUNICACIÓN	54
6.1.4. CLASE ALMACENAR SD	55
6.1.5. CLASE SENSOR NIVEL	56

6.1.6. CLASE SENSOR TEMPERATURA	56
6.1.7. CLASE SENSOR HUMEDAD	57
6.1.8. CLASE SENSOR FLUJO	57
6.1.9. CLASE ALMACEN	58
6.2. DEPENDENCIAS	59
7. ANEXOS.....	60
7.1. GLOSARIO	60
8. BIBLIOGRAFÍA	62

1. Introducción

1.1. 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.

1.2. 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.

WBS	Name	Start	Finish
1	Prototipo 2 Etapa	Jan 31	Mar 6
1.1	Construcción Base de Datos	Feb 3	Feb 4
1.1.1	Esquema de Base de Datos	Feb 3	Feb 3
1.1.2	Codificación Base de Datos	Feb 3	Feb 3
1.1.3	Pruebas Base de Datos	Feb 4	Feb 4
1.2	Análisis Servicio Web	Feb 4	Feb 5
1.2.1	Determinar Metodología de Trabajo	Feb 4	Feb 4
1.2.2	Esquema de Trabajo Servicio Web	Feb 5	Feb 5
1.3	Construcción Servicio Web	Feb 6	Feb 10
1.3.1	Codificación Servicio Web	Feb 6	Feb 10
1.4	Prueba Transferencia de Datos	Feb 10	Feb 14
1.4.1	Esquema de Pruebas	Feb 10	Feb 10
1.4.2	Ejecución de Pruebas	Feb 11	Feb 13
1.4.3	Recopilación información pruebas	Feb 11	Feb 13
1.4.4	Retroalimentación	Feb 14	Feb 14
1.5	Análisis Shield Ethernet Arduino	Feb 18	Feb 20
1.5.1	Reconocimiento de Hardware	Feb 18	Feb 18
1.5.2	Integración en Prototipo	Feb 19	Feb 20
1.5.2.1	Diagrama de Clase	Feb 19	Feb 19
1.5.2.2	Agregar shield en circuito	Feb 20	Feb 20
1.6	Transferencia Datos	Feb 21	Feb 26
1.6.1	Esquema de Pruebas	Feb 21	Feb 21
1.6.2	Ejecución de Pruebas	Feb 24	Feb 24
1.6.3	Recopilación información Pruebas	Feb 24	Feb 24
1.6.4	Retroalimentación	Feb 26	Feb 26
1.7	Pruebas sobre sistema	Feb 27	Feb 27
1.8	Pruebas de Trabajo en Prototipo 1	Jan 31	Feb 12
1.9	Construcción GUI	Feb 28	Mar 6

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 se 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	Oculto
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	Oculto
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.	Oculto

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.	Oculto
R.2.2	Escribir en Memoria	El sistema deberá registrar los datos que sean solicitados.	Oculto
R.2.3	Error en Memoria	El sistema deberá alertar sobre errores en el registro de datos.	Oculto
R.2.4	Leer Memoria	El sistema deberá permitir la lectura de información desde la memoria	Oculto
R.2.5	Crear	El sistema deberá permitir la creación de registros para almacenar datos	Oculto
R.2.6	Liberar Memoria	El sistema deberá lograr liberar memoria que sea solicitada	Oculto
R.2.7	Entregar Datos	El sistema deberá realizar la tarea de entrega de datos almacenados en memoria	Oculto

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.	Ocultas
R.3.2	Recibir Mensaje	El sistema deberá almacenar y mostrar el mensaje recibido por medio de funciones integradas	Ocultas
R.3.3	Borrar Mensaje	El sistema deberá permitir borrar el mensaje utilizado. (Requerimiento por memoria de Dispositivo)	Ocultas
R.3.4	Convertir Mensaje	El sistema deberá permitir la conversión del algún mensaje a numero para ahorrar espacio de memoria	Ocultas
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.	Ocultas
R.3.7	Setear Mensaje	El sistema deberá permitir Setear el mensaje para su posterior utilización al enviar datos.	Ocultas

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.	Ocultas
R.4.2	Leer mensaje	El sistema deberá permitir la lectura de un mensaje enviado por la conexión de red.	Ocultas
R.4.3	Eliminar Mensaje	El sistema deberá permitir borrar el mensaje utilizado. (Requerimiento por memoria de Dispositivo)	Ocultas
R.4.4	Procedimiento de Envío	El sistema deberá permitir asegurar el envío de un mensaje hacia el servidor	Ocultas
R.4.5	Envío por Red	El sistema deberá enviar un mensaje por la red establecida.	Ocultas
R.4.6	Error de Conexión	El sistema deberá alertar si el mensaje no ha logrado ser entregado al servidor	Ocultas

R.4.7	Enviar mensaje	El mensaje es enviado por esta función	Oculto
--------------	----------------	--	--------

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.	Oculto
R.5.2	Configurar parámetros	El sistema deberá lograr efectuar cambios sobre los módulos de comunicación para dirigir los mensajes	Oculto
R.5.3	Salir Modo Comando	El sistema deberá salir del modo comando de los módulos de comunicación para grabar configuración correctamente.	Oculto
R.5.4	Restablecer	El sistema deberá eliminar configuraciones previas del módulo de comunicación para dejarlo con configuración de fábrica	Oculto
R.5.5	Explorar red	El sistema deberá explorar la red que los módulos de comunicación crean para obtener los equipos disponibles.	Oculto
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.	Oculto
R.6.2	Obtener Medida	El sistema deberá entregar la medida solicitada a un sensor específico.	Oculto

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.	Oculto
R.7.3	Función sql	El sistema logrará almacenar información en una base de datos	Oculto
R.7.4	Función parsear	El sistema preparará la orden a ejecutar la función sql	Oculto

TABLA 7 LISTA DE FUNCIONES DEL SOFTWARE PARA EL SERVIDOR.

8.0	Ventana		
Ref. #	Función	Descripción	Categoría
R.8.1	Listar Puertos	El sistema listará 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 listará 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	Específico	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 lograr 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	Oculto	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	Oculto	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	Oculto	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	Oculto	Velocidad	El sistema lograra habilitar la memoria para almacenar información en menos de 2 segundos	Obligatoria
R.2.2	Escribir en Memoria	Oculto	Velocidad Comunicación		
R.2.3	Error en Memoria	Oculto	Velocidad Interfaz	El sistema podrá responder con alertas ante eventuales fallos en su memoria de almacenamiento	Obligatoria
R.2.4	Leer Memoria	Oculto	Velocidad	El sistema logrará adquirir datos almacenados en memoria en un periodo no mayor a medio segundo.	Obligatoria
R.2.5	Crear	Oculto	Velocidad	El sistema logrará crear espacios de memoria definidos para un conjunto de datos específicos	Obligatoria
R.2.6	Liberar Memoria	Oculto	Velocidad Rendimiento	El sistema tendrá la capacidad de liberar memoria que no sea de utilidad	Obligatoria
R.2.7	Entregar Datos	Oculto	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	Oculto	Comunicación Velocidad	El mensaje será enviado entre Nodo Receptor y Nodo Sensor para establecer un flujo	Obligatorio

				de trabajo	
R.3.2	Recibir Mensaje	Oculto	Comunicación Velocidad	El mensaje será recepcionado y confirmado para su posterior utilización	Obligatorio
R.3.3	Borrar Mensaje	Oculto	Velocidad Rendimiento	Los mensajes que sean utilizados logran ser borrados para optimizar el uso de memoria en placas de Arduino	Obligatorio
R.3.4	Convertir Mensaje	Oculto	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	Oculto	Rendimiento Comunicación Velocidad	El sistema logrará limpiar los buffer de entrada y salida para no acumular información inútil.	Opcional
R.3.7	Setear Mensaje	Oculto	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	Oculto	Rendimiento Velocidad Conectividad	El sistema podrá conectarse a un red LAN para comunicaciones por esta.	Obligatorio
R.4.2	Leer mensaje	Oculto	Conectividad	El sistema deberá	Obligatoria

			Velocidad	adquirir los mensajes que sean enviados por la red en menos de 1 segundo	
R.4.3	Eliminar Mensaje	Oculto	Rendimiento Velocidad	El sistema debe ser capaz de eliminar mensajes que no sean de utilidad	Obligatorio
R.4.4	Procedimiento de Envío	Oculto	Rendimiento Velocidad Conectividad	Un procedimiento simplificado para poder lograr la comunicación por medio de una red	Opcional
R.4.5	Envío por Red	Oculto	Conectividad Velocidad	El sistema enviará por medio de la red a un dirección específica un mensaje para su procesamiento	Obligatorio
R.4.6	Error de Conexión	Oculto	Rendimiento Velocidad	El sistema advertirá si algún fallo es percibido en el envío de información por la red	Obligatorio
R.4.7	Enviar Mensaje	Oculto	Conectividad Velocidad	El sistema realiza la ejecución del envío 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	Oculto	Específico Comunicación Velocidad	El sistema permitirá configurar los módulos de comunicación durante la ejecución de solicitudes	Obligatoria
R.5.2	Configurar parámetros	Oculto	Específico Velocidad	El sistema logrará configurar parámetros específicos de los módulos de comunicación	Obligatoria

R.5.3	Salir Modo Comando	Oculto	Velocidad Rendimiento	El sistema asegurará que las configuraciones sean guardadas correctamente	Obligatoria
R.5.4	Restablecer	Oculto	Rendimiento	El sistema podrá restablecer las configuraciones de los módulos de comunicación.	Opcional
R.5.5	Explorar red	Oculto	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	Oculto	Específico	El sistema permitirá la obtención de un valor métrico desde un valor analógico de sensor	Obligatoria
R.6.2	Obtener Medida	Oculto	Específico	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	Oculto	Conectividad Velocidad	Los clientes conectados serán independientes entre sí.	Obligatoria
R.7.3	Función sql	Oculto	Conectividad	La información que provenga desde el cliente será transferida hacia una Base de Datos	Obligatoria
R.7.4	Función parsear	Oculto	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	Obligatoria


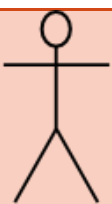
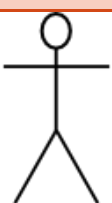
				conectan los dispositivos arduino nodo receptores	
R.8.2	Obtener Equipos	Evidente	Interfaz	Véase 5.6	Obligatoria
R.8.3	Listar Sensores	Evidente	Interfaz	El sistema listará los sensores pertenecientes a un dispositivos previa consulta a la Base de Datos	Obligatoria

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 se describen a continuación.

Actores	
Actor	Descripción
 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
 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.
 Arduino	Este actor se encargará de ejecutar las acciones destinadas por el software por medio de sus componentes de hardware.



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 almacenar 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	Envío 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 desconecta

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 interacción 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.
--	---

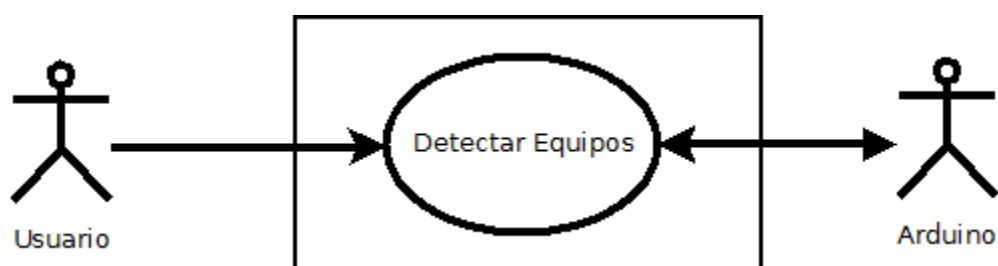
Cursos Alternos

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.

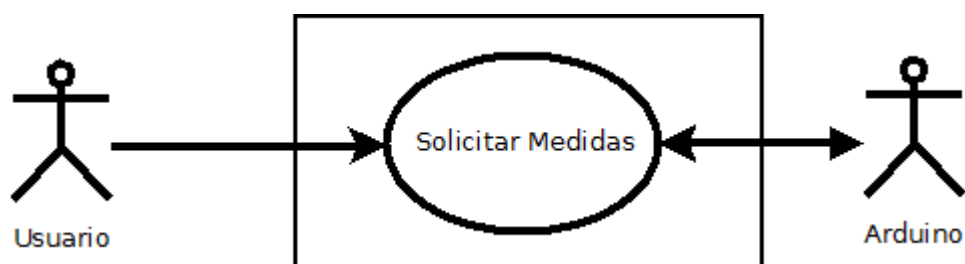
3.4.2. Diagrama de Caso de Uso

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

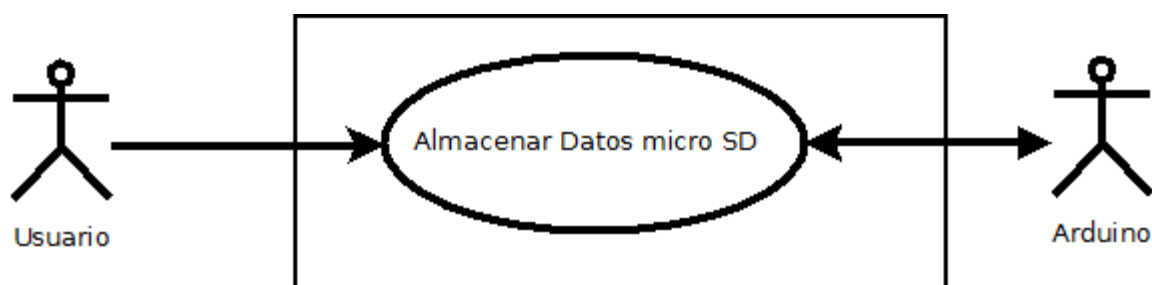
C01.- Detectar Equipos



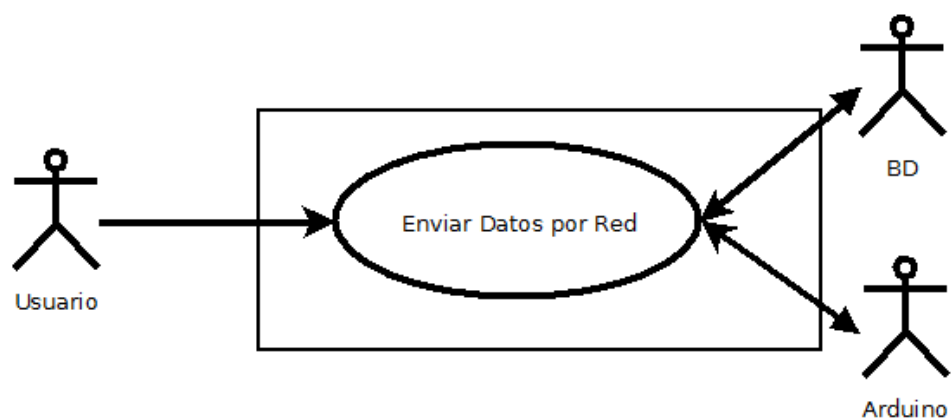
C02.- Solicitar Medidas



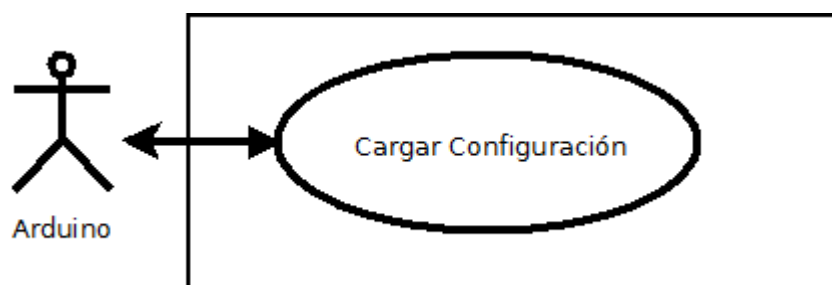
C03.- Almacenar Datos Micro SD



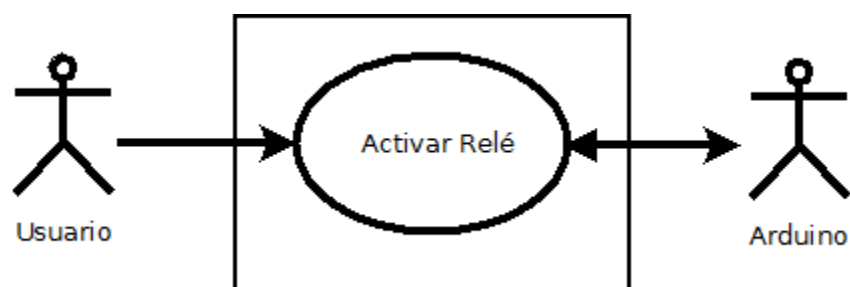
C04.- Enviar Datos por Red



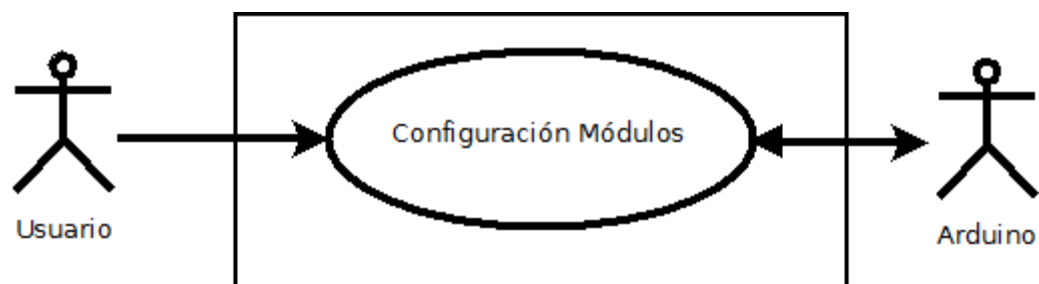
C05.- Cargar Configuración



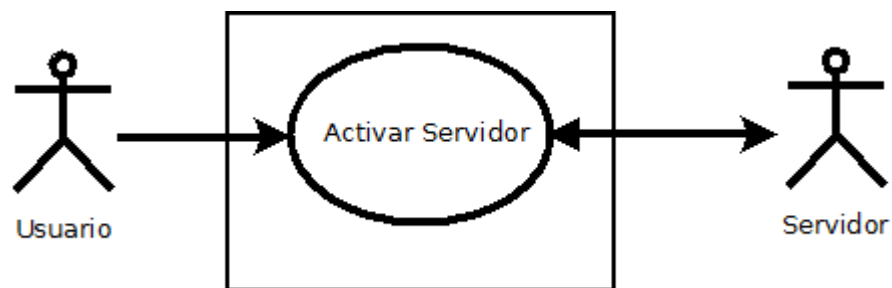
C06.- Activar Relé



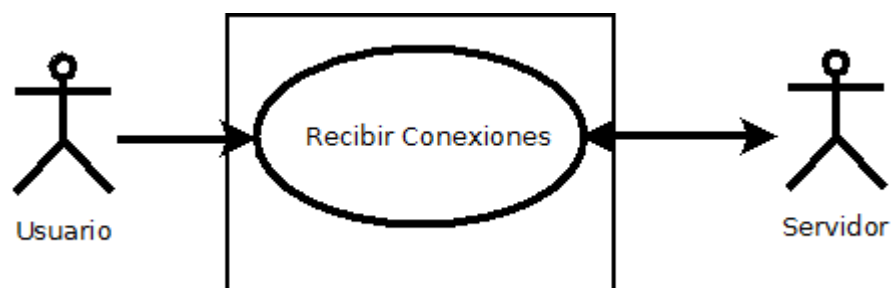
C07.- Configurar Módulos



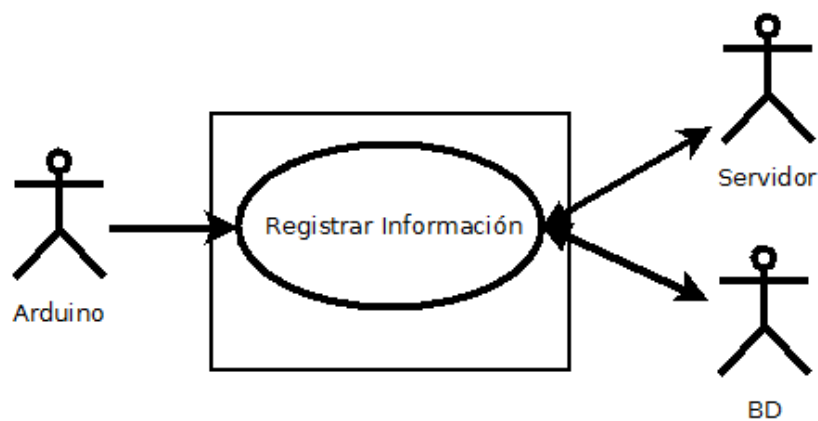
C08.- Activar Servidor



C09.- Recibir Conexiones



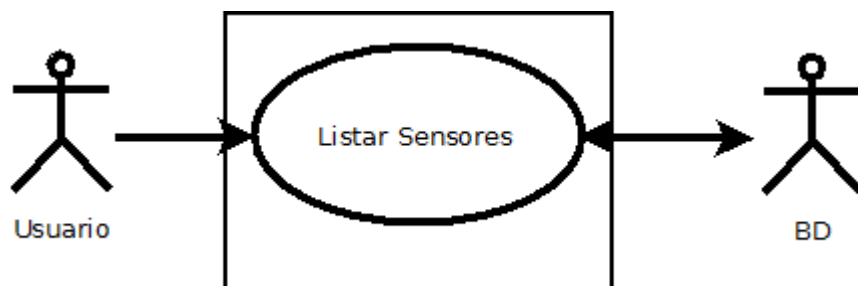
C10.- Registrar Información



C11.- Ver Puertos



C12.- Listar Sensores



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

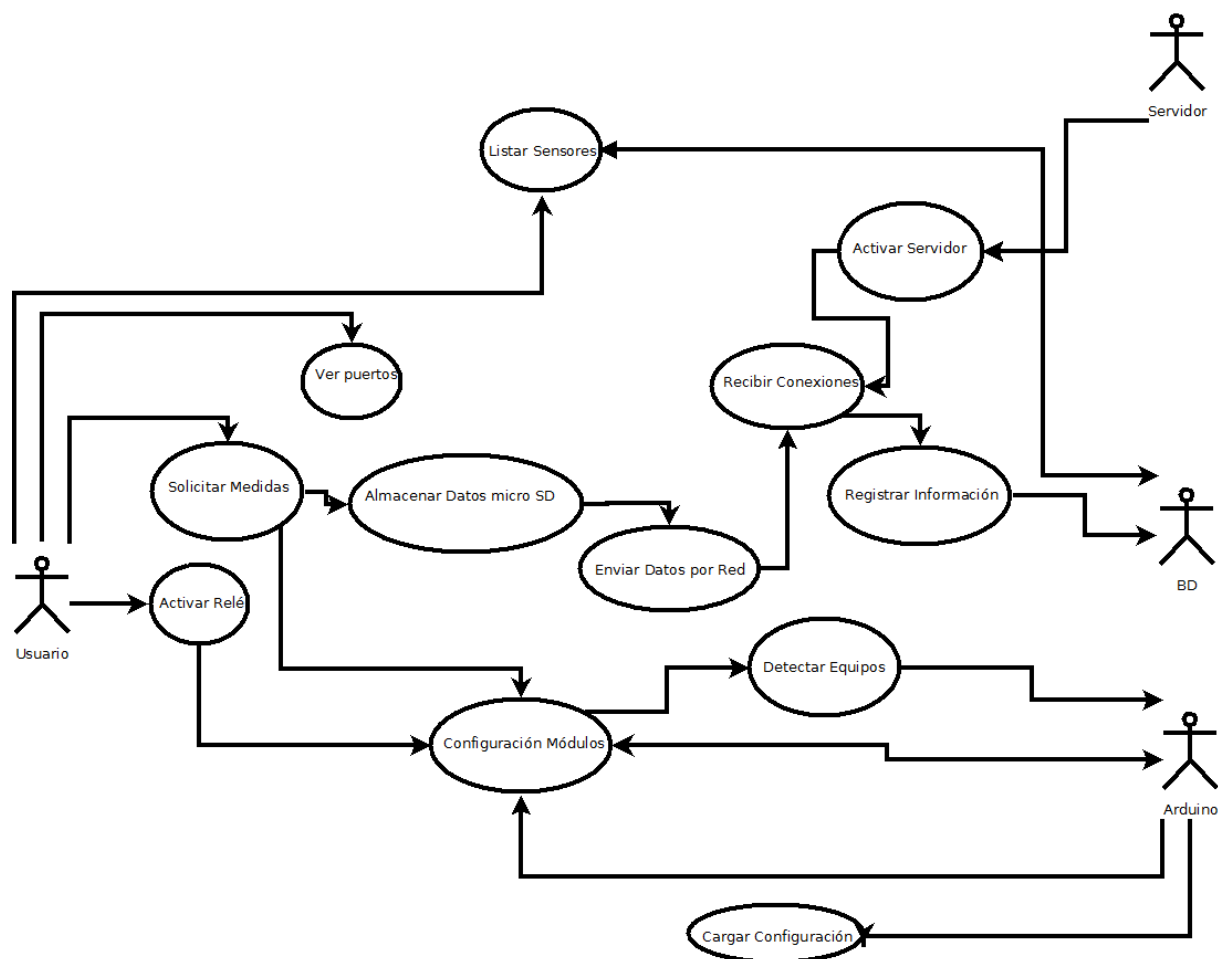


FIGURA 2: ESQUEMA GENERAL DE ACTORES Y CASOS DE USO.

3.4.3. Modelo Conceptual

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

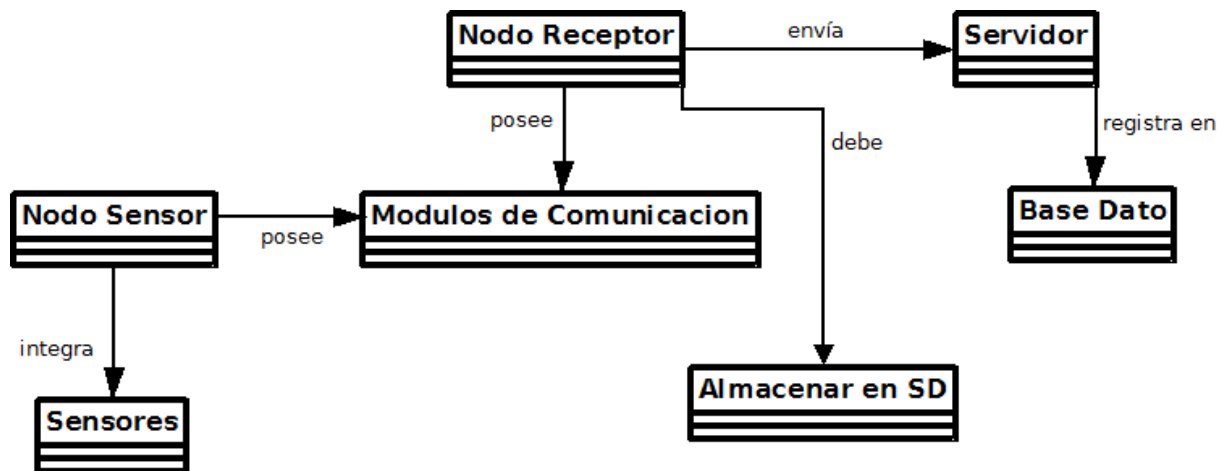
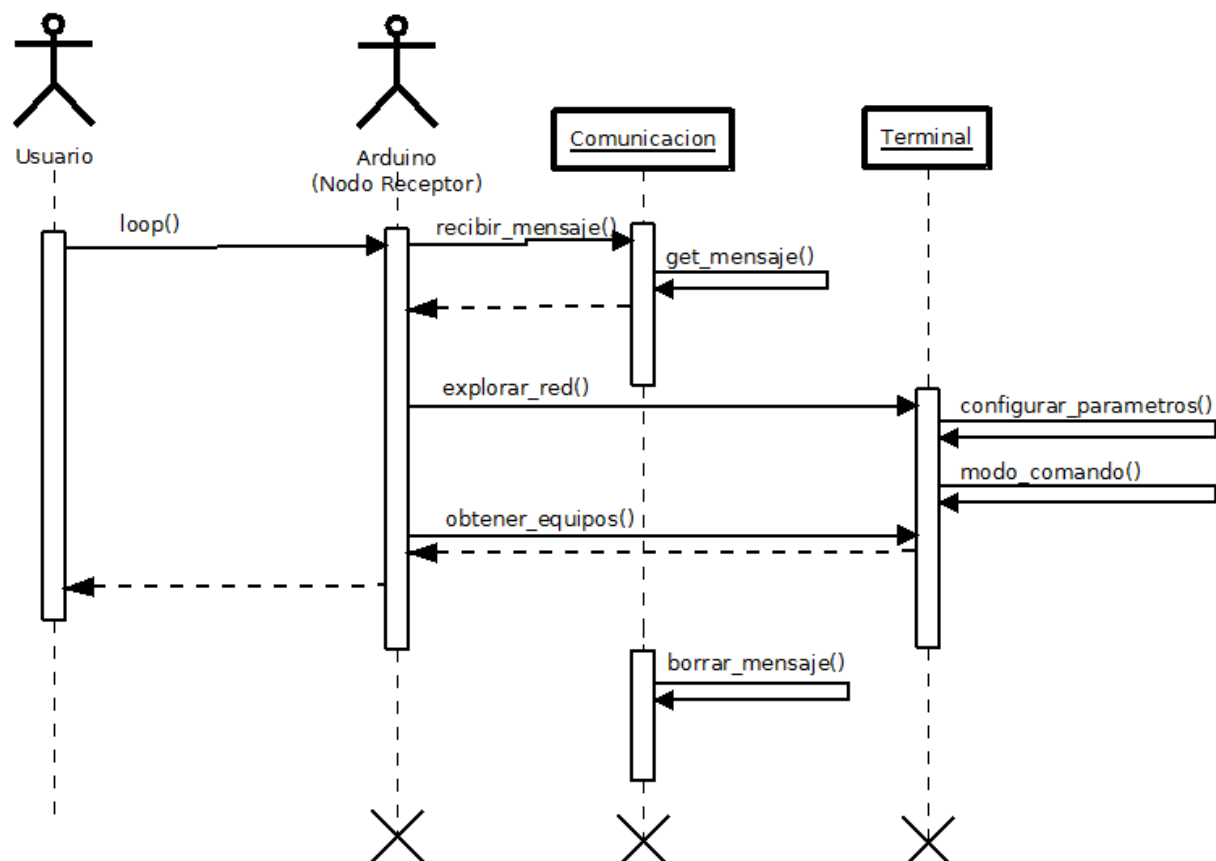


FIGURA 3: MODELO CONCEPTUAL DEL SOFTWARE.

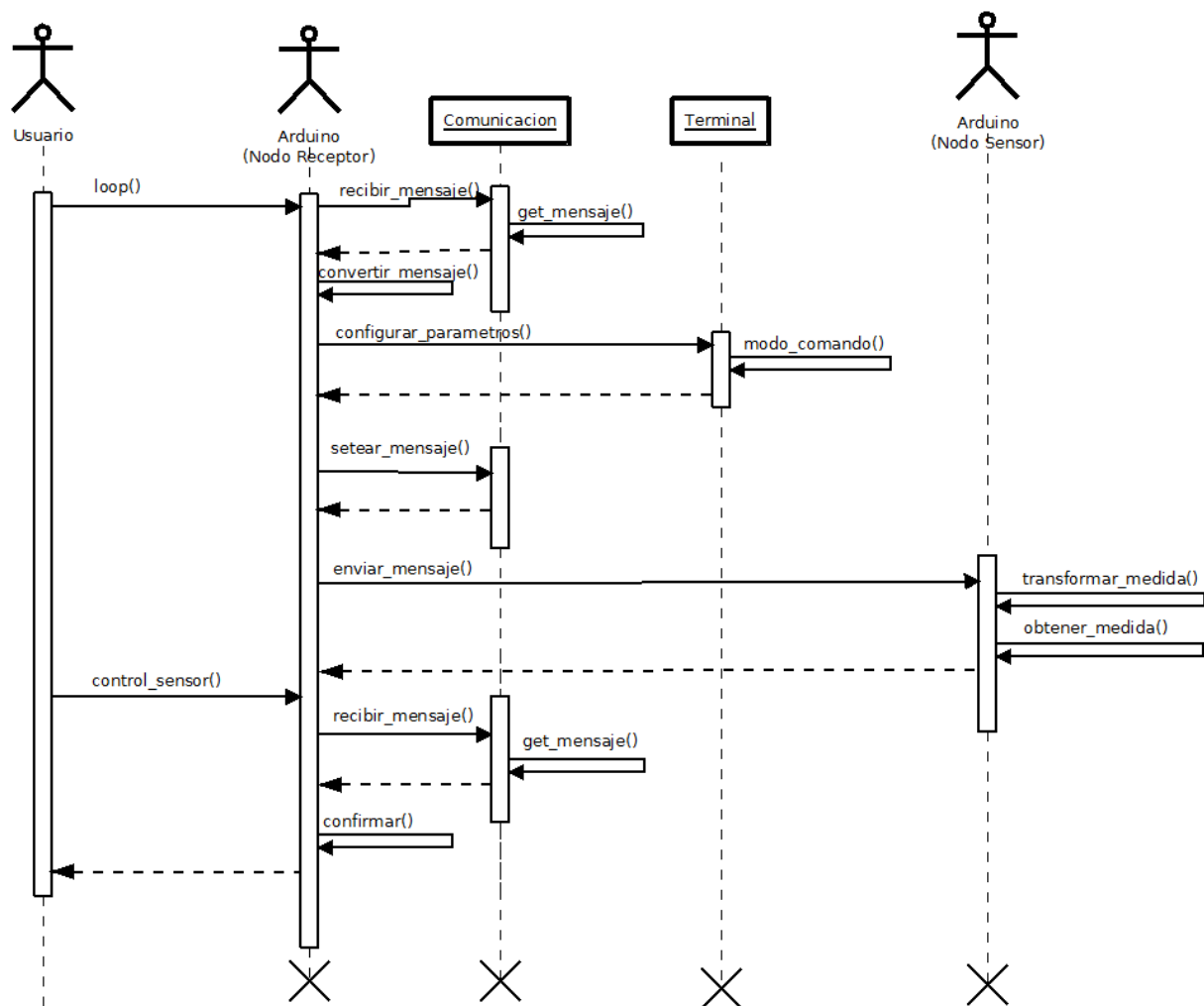
3.4.4. Diagrama de Secuencia o Colaboración

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.

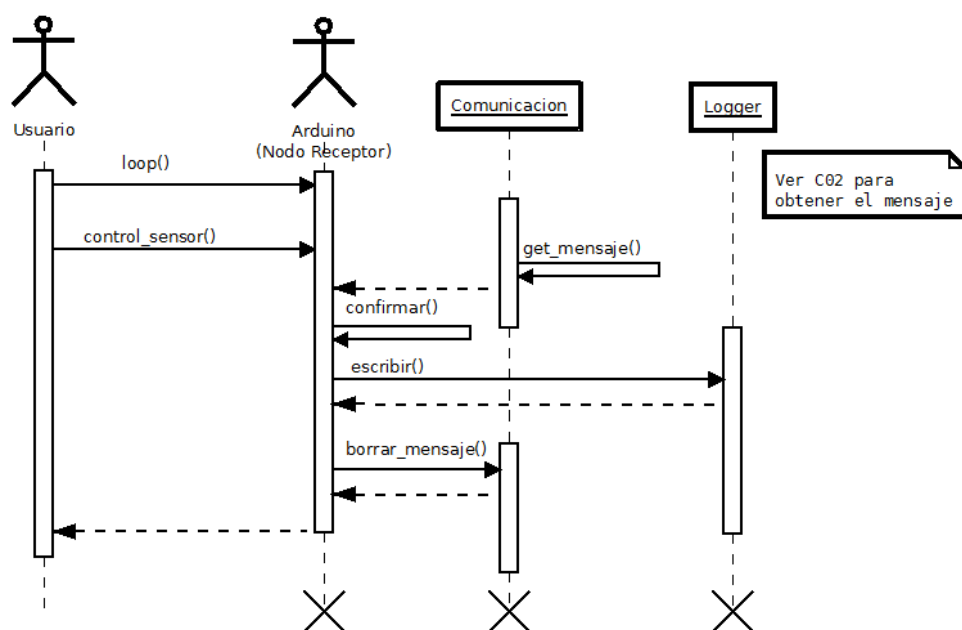
C01.- Detectar Equipos



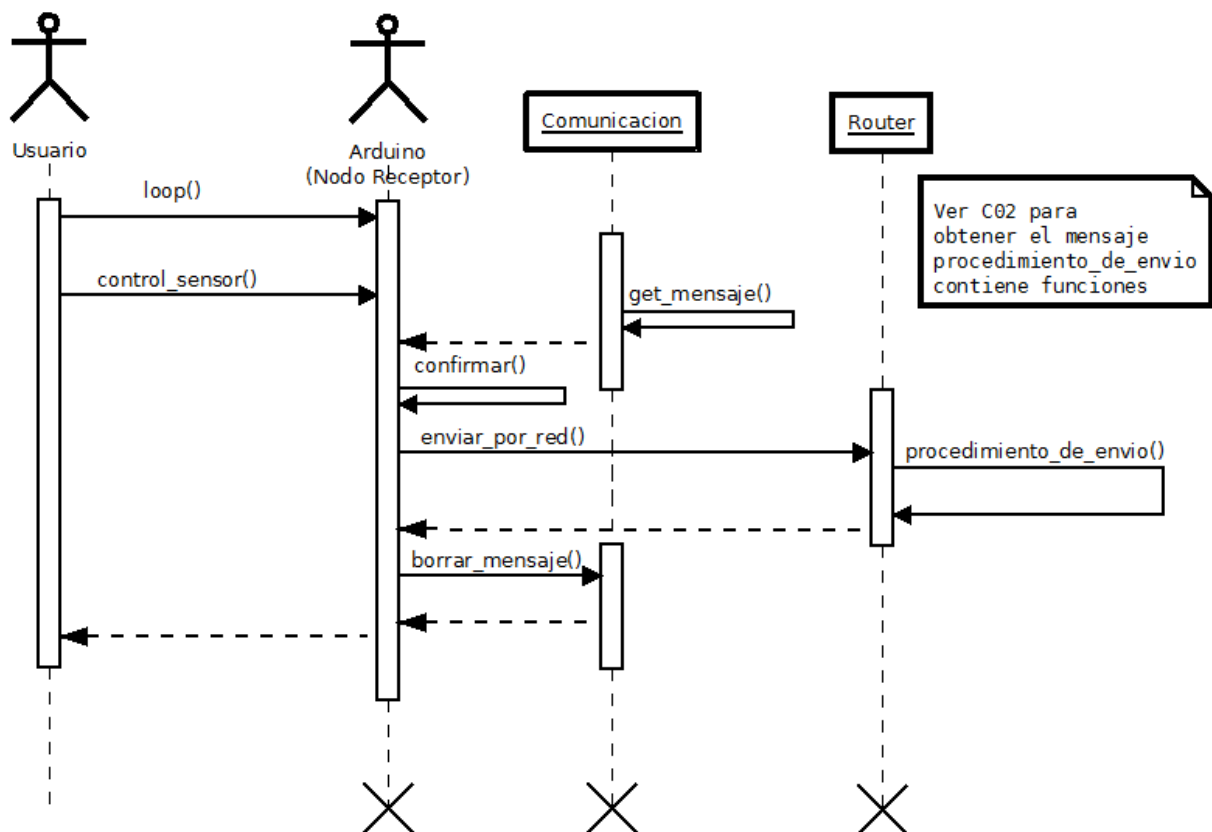
C02.- Solicitar Medidas



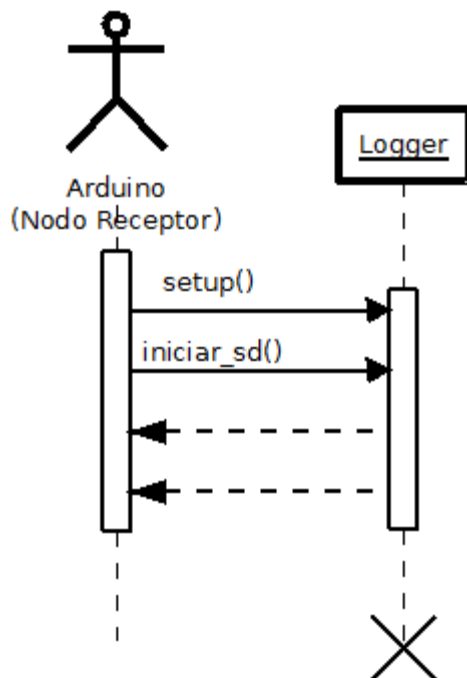
C03.-Almacenar Datos Micro SD



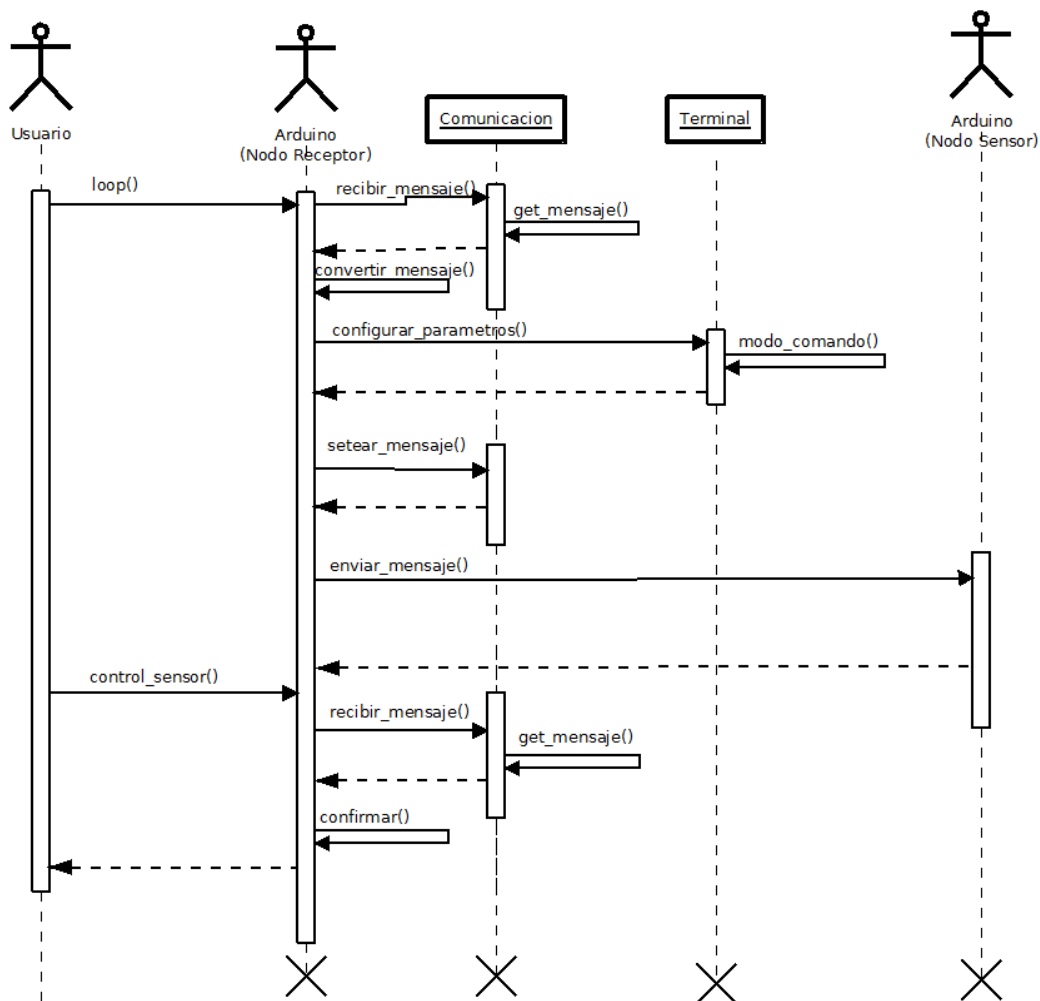
C04.- Enviar Datos por Red



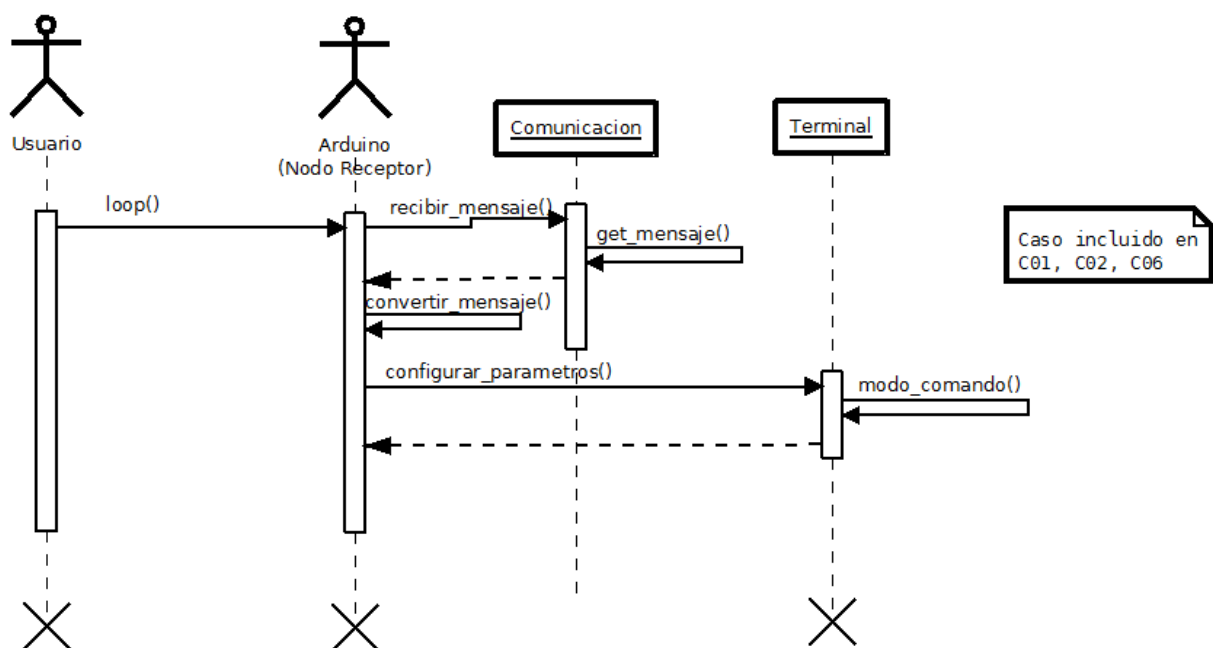
C05.- Cargar Configuración



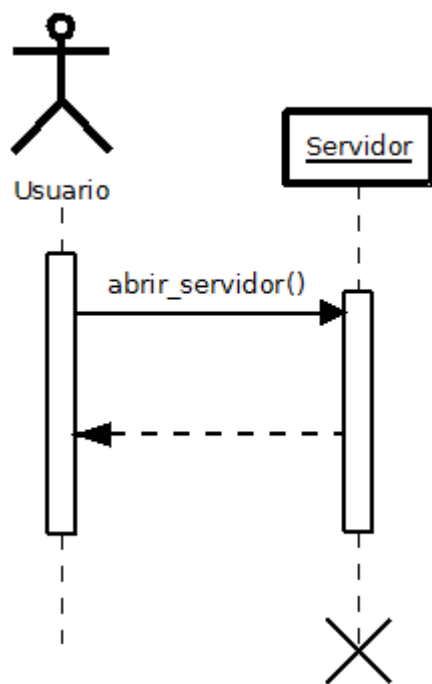
C06.- Activar Relé



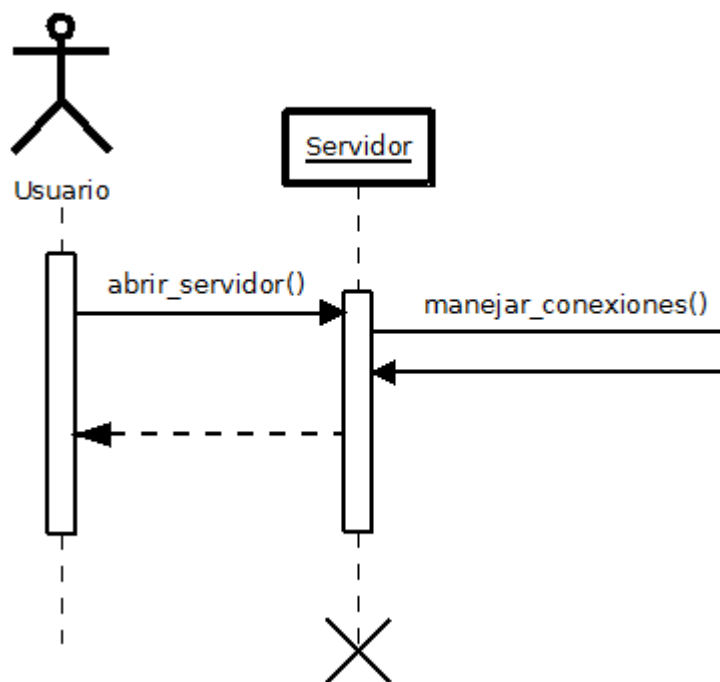
C07.- Configuración Módulos



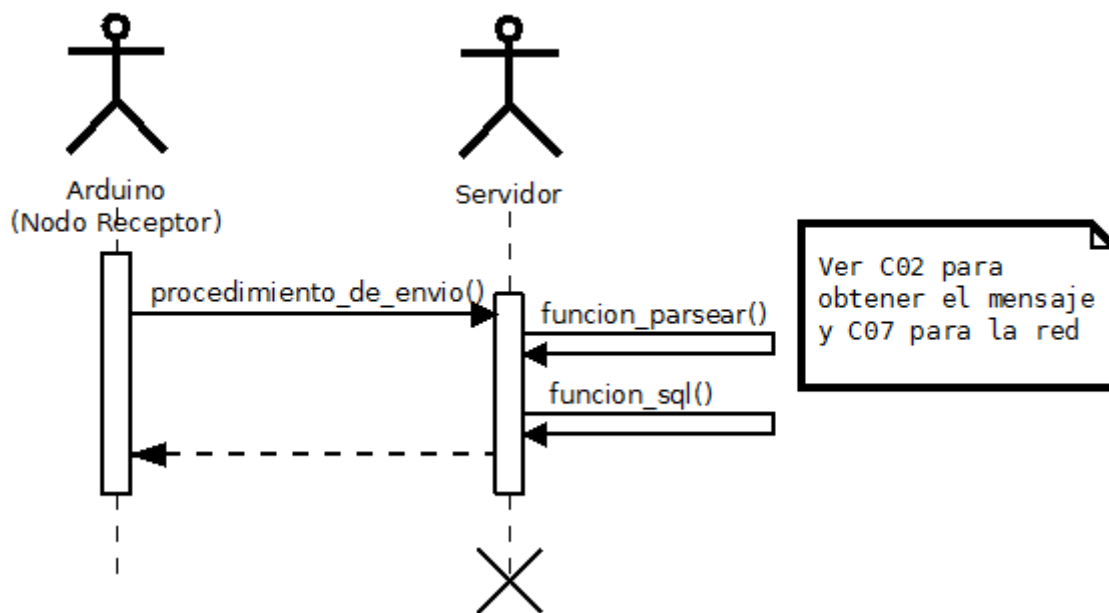
C08.- Activar Servidor



C09.- Recibir Conexiones



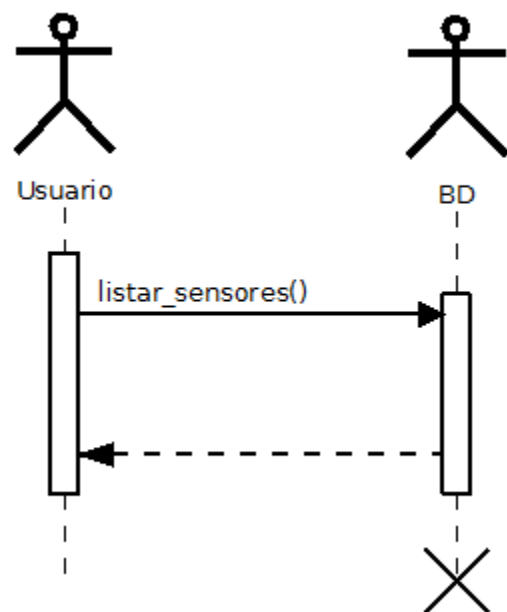
C10.- Registrar Información



C11.- Ver Puertos

Para el caso del diagrama de interacción 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.

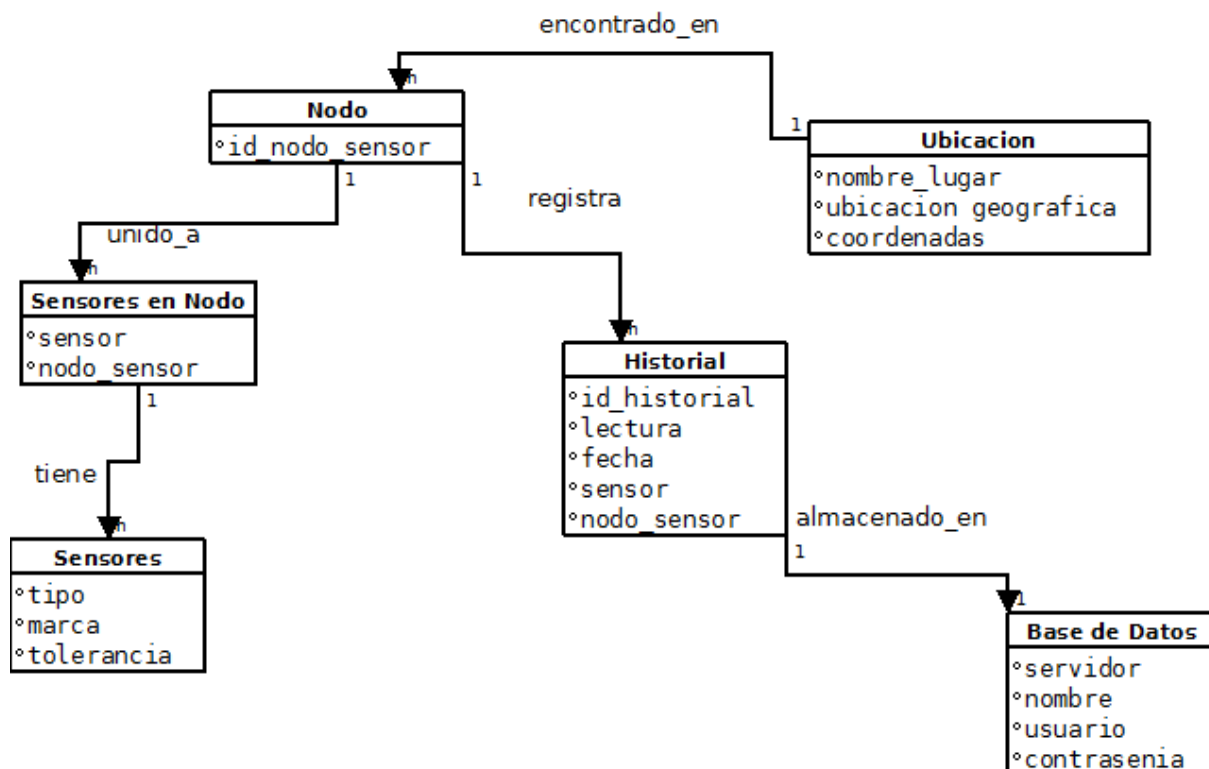


FIGURA 4: MODELO DE DOMINIO, ENTIDADES AGRUPADAS EN PAQUETES DE UNIDAD CONCEPTUAL JUNTO A SUS DEPENDENCIAS.

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

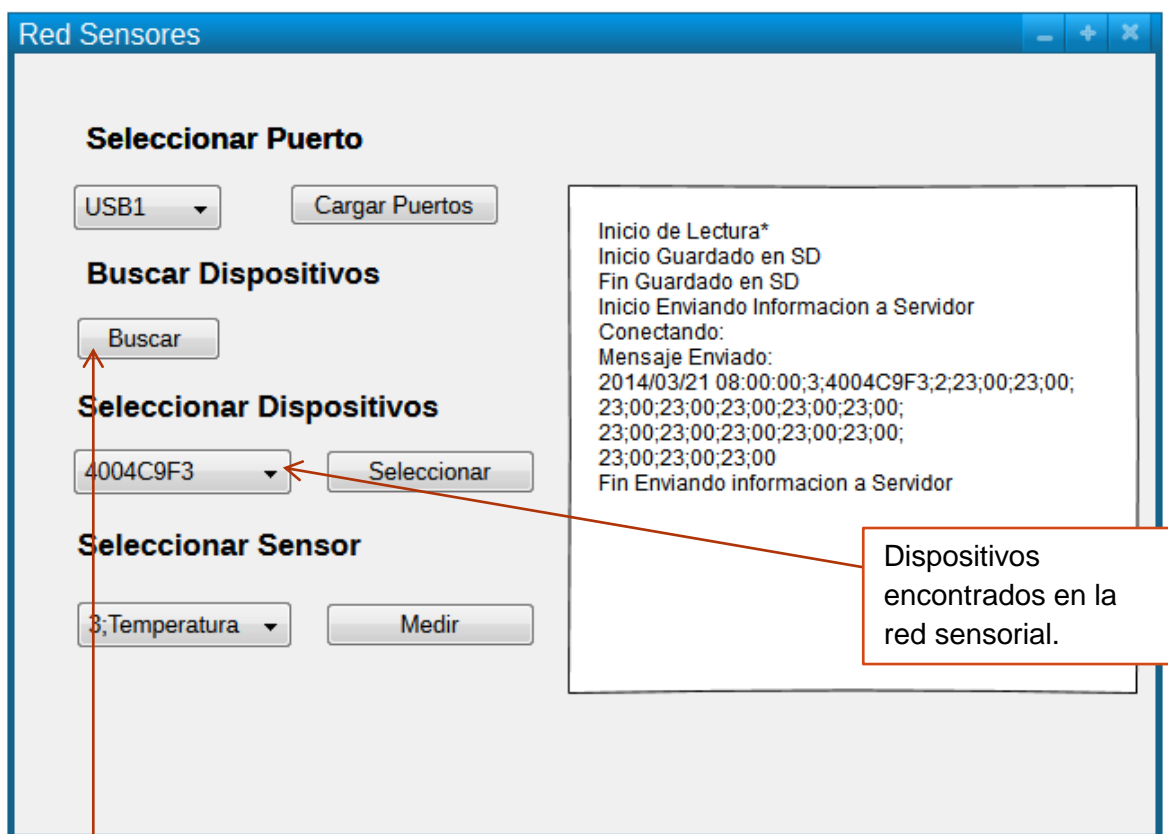
En esta sección se presentan los prototipos del sistema, su utilización y descripción según los casos de uso descritos en la sección 3.4.

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.



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

Dispositivos encontrados en la red sensorial.

C02 Solicitar Medidas

Red Sensores

Seleccionar Puerto

USB1 Cargar Puertos

Buscar Dispositivos

Buscar

Seleccionar Dispositivos

4004C9F3 Seleccionar

Seleccionar Sensor

3;Temperatura Medir

Inicio de Lectura*
Inicio Guardado en SD
Fin Guardado en SD
Inicio Enviando Informacion a Servidor
Conectando:
Mensaje Enviado:
2014/03/21 08:00:00;3;4004C9F3;2;23;00;23;00;
23;00;23;00;23;00;23;00;23;00;
23;00;23;00;23;00;23;00;23;00;
23;00;23;00;23;00
Fin Enviando informacion a Servidor

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

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

C08 Activar Servidor

Red Sensores

Ingreso IP 192.168.1.33 **Ingreso Puerto** 3010 Encender Servidor

Conexion Realizada
Inicio Recepcion

Indicación de la activación

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

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.

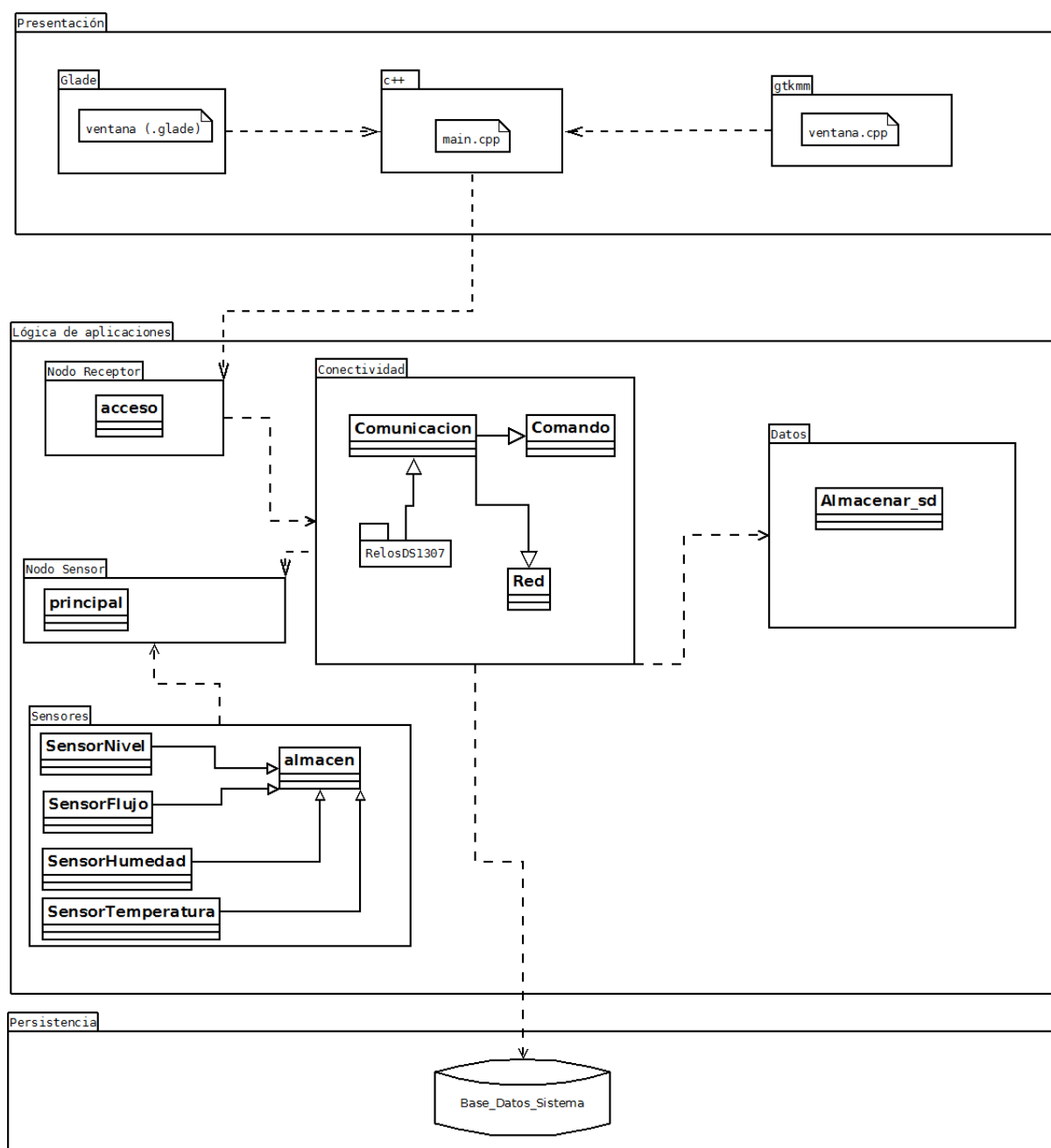


FIGURA 5: MODELO DE SOFTWARE INICIAL DEL SISTEMA CORRESPONDIENTE AL MODELO DE TRES CAPAS, SE PRESENTAN LOS PAQUETES Y CLASES LAS CUALES SE USAN COMO BASE DEL SOFTWARE.

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.

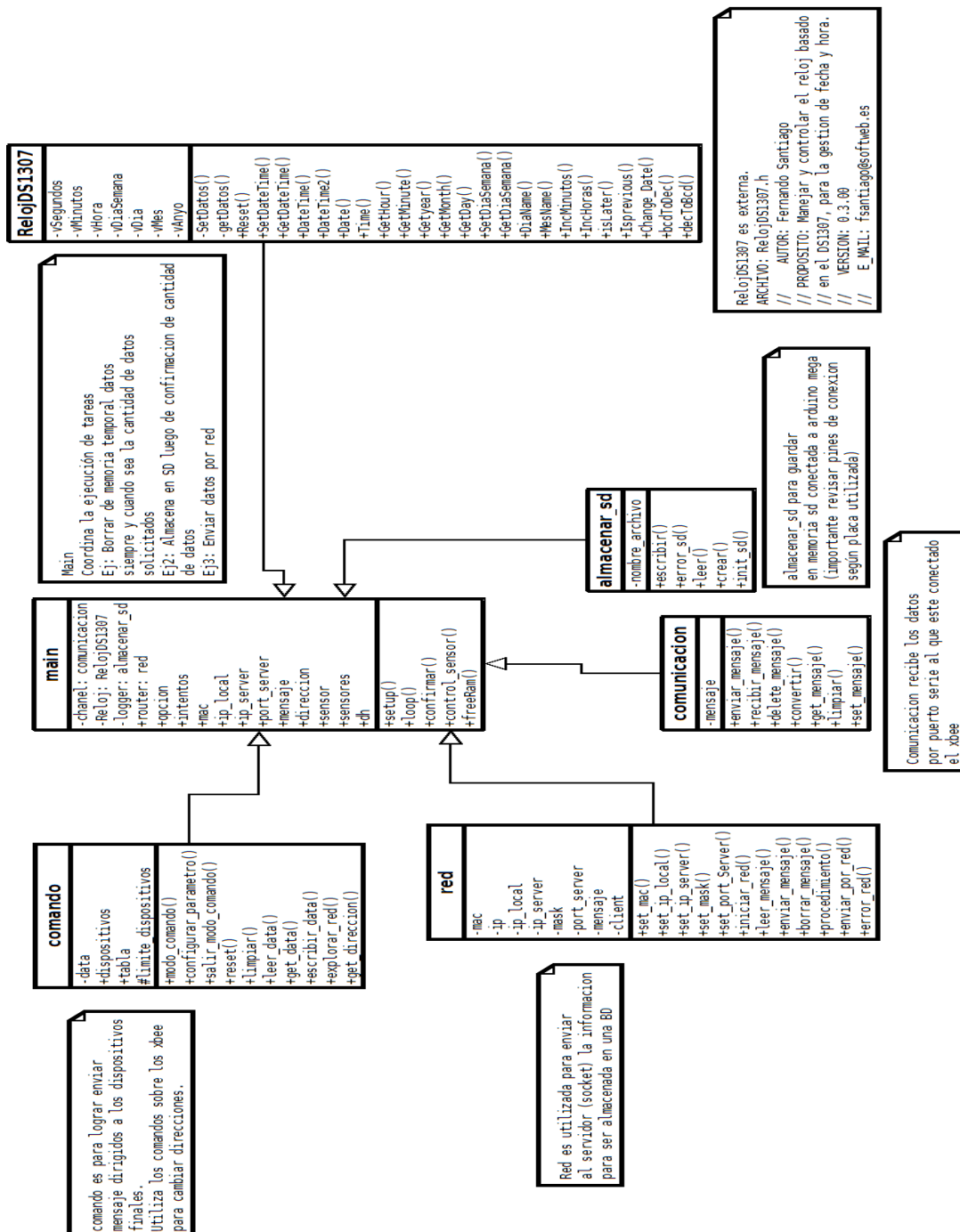


FIGURA 6: DIAGRAMA DE CLASES DE DISEÑO DEL SOFTWARE, JUNTO CON SUS ATRIBUTOS Y SUS RESPECTIVOS MÉTODOS PARA HARDWARE NODO RECEPTOR.

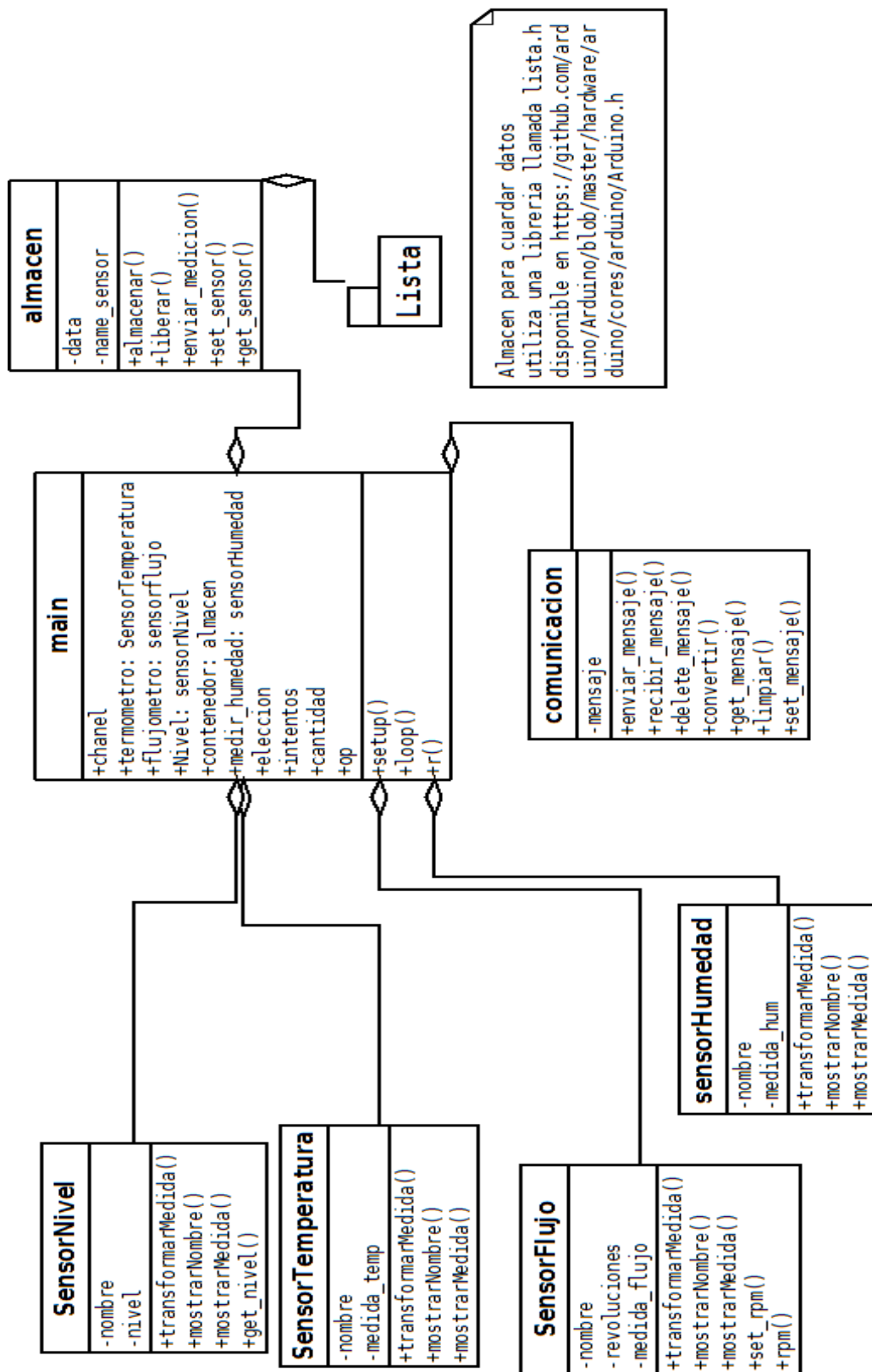
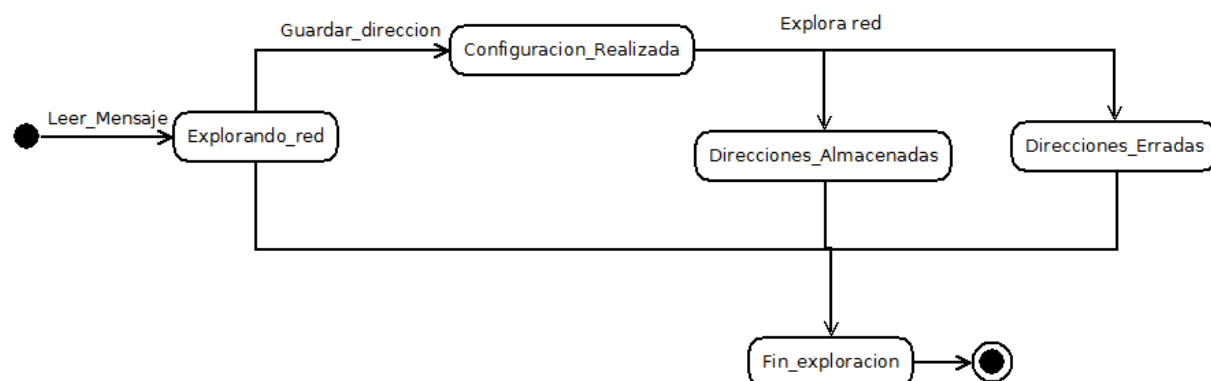


FIGURA 7 DIAGRAMA DE CLASES DE DISEÑO DEL SOFTWARE, JUNTO CON SUS ATRIBUTOS Y SUS RESPECTIVOS MÉTODOS PARA HARDWARE NODO SENSOR.

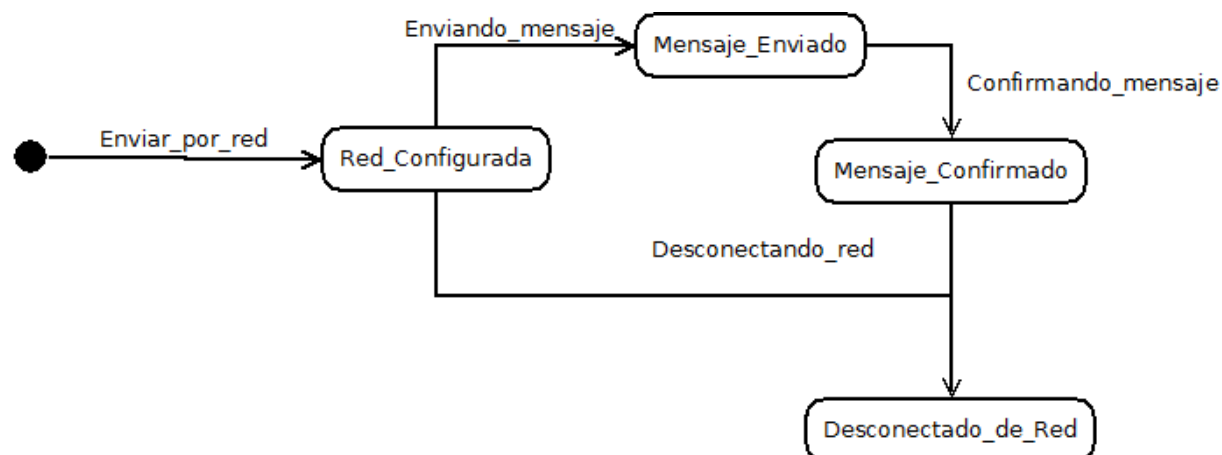
5.1.4. Diagramas de Estados

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

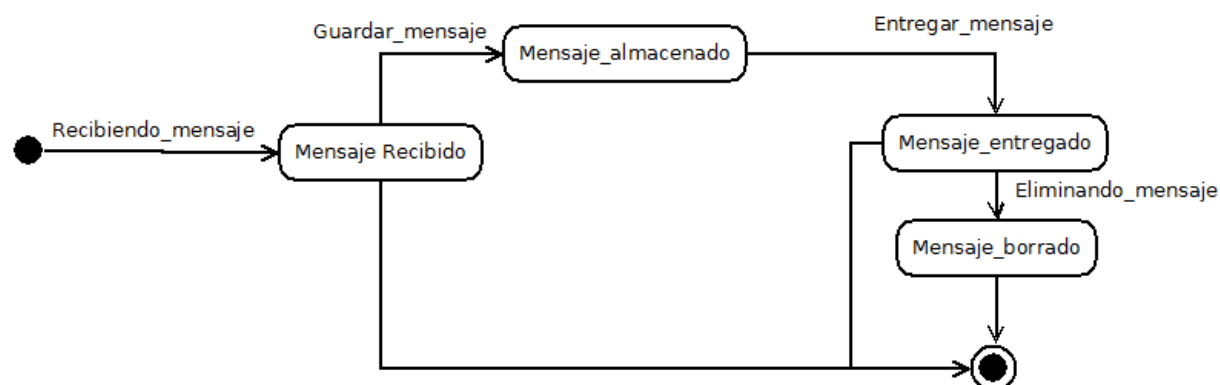
Clase.-Comando



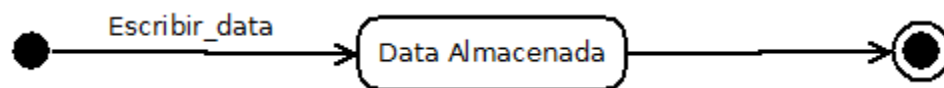
Clase.-Red



Clase.-Comunicación

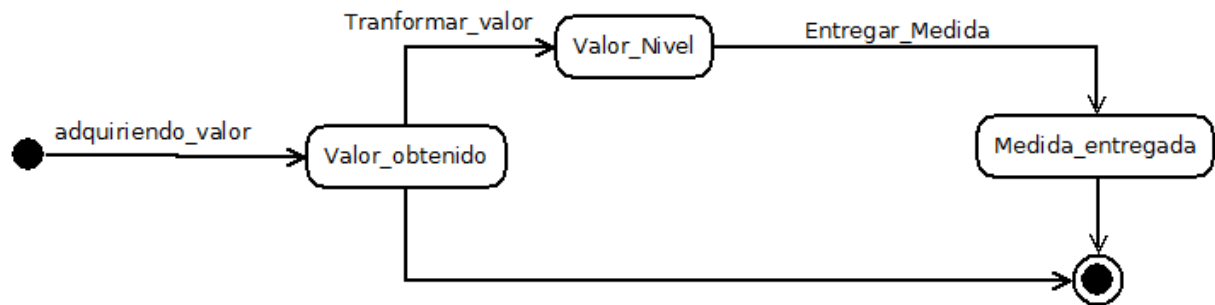


Clase.-Almacenar SD

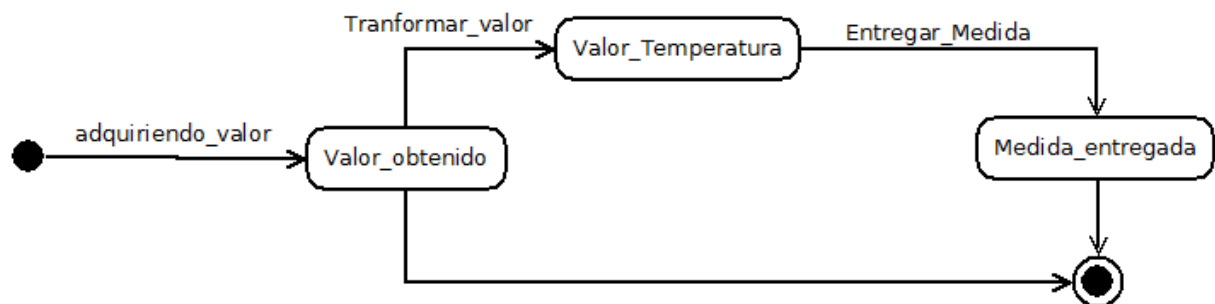


Para hardware nodo sensor

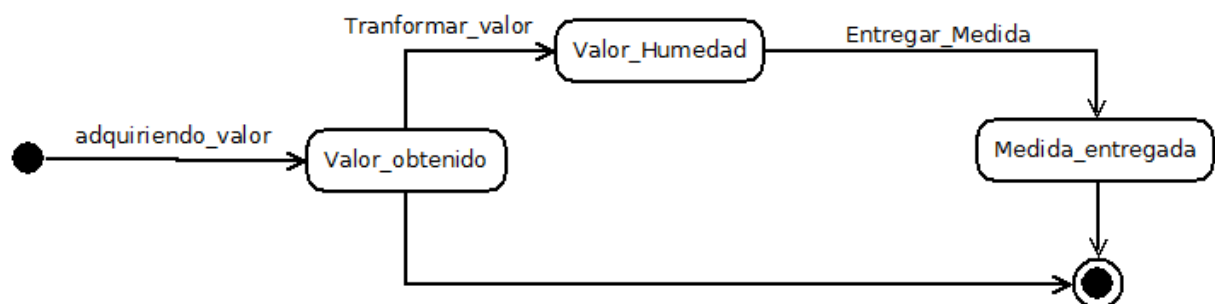
Clase.-Sensor Nivel



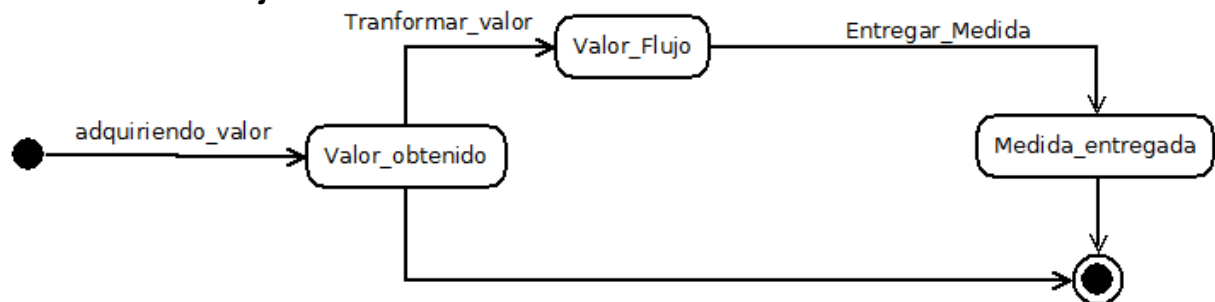
Clase.-Sensor Temperatura



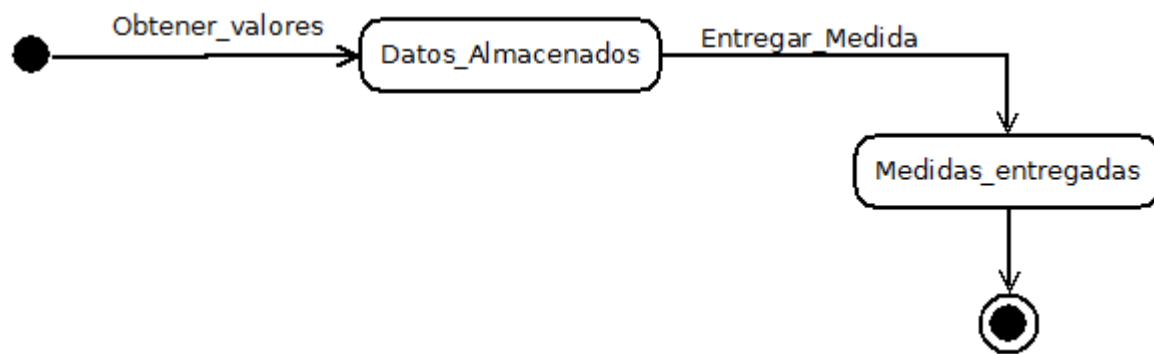
Clase.-Sensor Humedad



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.practicadesoftware.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://decea.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/>