**Sensores** - **Grupo 7**

**Descripción de la Arquitectura**

**Versión 2.4**

**Historia de revisiones**

|  |  |  |  |
| --- | --- | --- | --- |
| Fecha | Versión | Descripción | Autor |
| 29/08/14 | 1.0 | Creación del documento | Diego Serra |
| 30/08/14 | 1.1 | Vista del modelo casos de uso | Diego Serra |
| 31/08/14 | 1.2 | Diagrama de Distribución | Diego Serra |
| 05/09/14 | 1.3 | Descomposición en subsistemas + Correcciones | Diego Serra |
| 10/09/14 | 2.0 | Correcciones | Diego Serra |
| 19/09/14 | 2.1 | Correcciones | Diego Serra |
| 27/09/2014 | 2.2 | Correcciones de CU | Diego Serra |
| 05/10/2014 | 2.3 | Correcciones SQA | Oscar Muñoz |
| 10/10/2014 | 2.4 | Diagrama de trazabilidad | Diego Serra |
| 20/10/2014 | 3.0 | Correcciones detectadas por verificación | Diego Serra |

Contenido

[1. Introducción 5](#_Toc400284886)

[1.1. Propósito 5](#_Toc400284887)

[1.2. Alcance 5](#_Toc400284888)

[1.3. Definiciones, siglas y abreviaturas. 5](#_Toc400284889)

[1.4. Referencias 5](#_Toc400284890)

[1.5. Visión general 5](#_Toc400284891)

[2. Vista del Modelo de Casos de Uso 6](#_Toc400284892)

[2.1. Diagrama de Casos de Uso relevantes a la Arquitectura 6](#_Toc400284893)

[2.2. Casos de Uso relevantes a la Arquitectura 7](#_Toc400284894)

[2.2.1. Alta nuevo tipo de sensor 7](#_Toc400284895)

[2.2.2. Login/Logout 7](#_Toc400284896)

[2.2.3. Alta alarma 7](#_Toc400284897)

[2.2.4. Visualizar capas de una red 7](#_Toc400284898)

[2.2.5. Visualización de interpolado en el mapa 7](#_Toc400284899)

[2.2.6. Alta Usuario (Agrónomo) 7](#_Toc400284900)

[2.2.7. Alta Usuario (Técnico) 8](#_Toc400284901)

[2.2.8. Alta red 8](#_Toc400284902)

[2.2.9. Modificar Red 8](#_Toc400284903)

[2.2.10. Ingresar mote a red 8](#_Toc400284904)

[2.2.11. Recibir Motes 8](#_Toc400284905)

[2.2.12. Asignar agrónomo a red 8](#_Toc400284906)

[2.2.13. Recibir Medición + Verificar Alarma +Notificar 8](#_Toc400284907)

[2.2.14. Recibir Padre 9](#_Toc400284908)

[2.2.15. Generar árbol de rutas de paquetes 9](#_Toc400284909)

[2.2.16. Cargar capas para el mapa 9](#_Toc400284910)

[2.2.17. Visualizar coloreo de motes 9](#_Toc400284911)

[2.2.18. Visualizar gráficas 9](#_Toc400284912)

[2.2.19. Visualizar árbol de rutas 9](#_Toc400284913)

[3. Trazabilidad desde el Modelo de Casos de Uso al Modelo de Diseño 10](#_Toc400284914)

[3.1. Diagramas de paquetes 10](#_Toc400284915)

[3.1.1. Alta nuevo tipo de sensor 10](#_Toc400284916)

[3.1.2. Login/Logout 10](#_Toc400284917)

[3.1.3. Alta alarma. 10](#_Toc400284918)

[3.1.4. Visualizar gráficas 10](#_Toc400284919)

[3.1.5. Visualizar coloreo de motes 11](#_Toc400284920)

[3.1.6. Visualizar capas de una red 11](#_Toc400284921)

[3.1.7. Visualización de interpolado en el mapa 12](#_Toc400284922)

[3.1.8. Alta Usuario (Técnico/Agrónomo) 12](#_Toc400284923)

[3.1.9. Alta red 12](#_Toc400284924)

[3.1.10. Modificar Red 12](#_Toc400284925)

[3.1.11. Ingresar mote a red 13](#_Toc400284926)

[3.1.12. Recibir Motes 13](#_Toc400284927)

[3.1.13. Asignar agrónomo a red 13](#_Toc400284928)

[3.1.14. Verificar Alarma 13](#_Toc400284929)

[3.1.15. Recibir Medición 14](#_Toc400284930)

[3.1.16. Recibir Padre 14](#_Toc400284931)

[3.1.17. Generar árbol de rutas de paquetes 14](#_Toc400284932)

[3.1.18. Cargar capas para el mapa 14](#_Toc400284933)

[4. Vista del Modelo de Diseño 15](#_Toc400284934)

[4.1. Descomposición en subsistemas 15](#_Toc400284935)

[4.1.1. Servidor de Aplicaciones 15](#_Toc400284936)

[4.2. Subsistemas específicos 15](#_Toc400284937)

[4.2.1. Core Logic 15](#_Toc400284938)

[4.2.2. Core Data Access 15](#_Toc400284939)

[4.2.3. ExternalComunicationMaps 15](#_Toc400284940)

[4.2.4. ExternalComunication Network 15](#_Toc400284941)

[4.2.5. Web Services 15](#_Toc400284942)

[4.3. Daemon 16](#_Toc400284943)

[4.4. Subsistemas específicos 16](#_Toc400284944)

[4.4.1. Network Comunication 16](#_Toc400284945)

[4.4.2. DaemonLogic 16](#_Toc400284946)

[4.4.3. Daemon Data Access 16](#_Toc400284947)

[4.4.4. Web Services 16](#_Toc400284948)

[4.5. Servidor Web 17](#_Toc400284949)

[4.6. Subsistemas específicos 17](#_Toc400284950)

[4.6.1. Web 17](#_Toc400284951)

[5. Trazabilidad desde el Modelo de Diseño al Modelo de Implementación 17](#_Toc400284952)

[6. Vista del Modelo de Implementación 17](#_Toc400284953)

[6.1. Manejo de errores 17](#_Toc400284954)

[7. Vista del Modelo de Distribución 18](#_Toc400284955)

[7.1. Diagrama de Distribución 18](#_Toc400284956)

[7.2. Nodos 18](#_Toc400284957)

[7.2.1. Cliente Web 18](#_Toc400284958)

[7.2.2. Servidor de Aplicaciones 18](#_Toc400284959)

[7.2.3. Daemon 19](#_Toc400284960)

[7.2.4. Servidor Web 19](#_Toc400284961)

[7.2.5. Base de datos PostGIS 19](#_Toc400284962)

[7.2.6. Base de datos PostgreSQL 19](#_Toc400284963)

[7.3. Conexiones 19](#_Toc400284964)

[7.3.1. Cliente Web – Servidor Web 19](#_Toc400284965)

[7.3.2. Servidor de Aplicaciones – Base de Datos Postgis 19](#_Toc400284966)

[7.3.3. Daemon – Base de Datos PostgreSQL 19](#_Toc400284967)

[7.3.4. Servidor de Aplicaciones – Base de Datos PostgreSQL 20](#_Toc400284968)

[7.3.5. Servidor de Aplicaciones – Daemon 20](#_Toc400284969)

[7.3.6. Servidor Web – Servidor de Aplicaciones 20](#_Toc400284970)

1. Introducción
   1. Propósito

Este documento proporciona una apreciación global y comprensible de la arquitectura del sistema usando diferentes puntos de vista para mostrar distintos aspectos del sistema. Intenta capturar y llegar a las decisiones de arquitectura críticas que han sido hechas en el sistema.

También sirve como base para el diseño del sistema y será presentado al cliente para que comprenda tanto aspectos globales como específicos del diseño del sistema.

* 1. Alcance

El siguiente documento comprende aspectos de diseño del sistema y servirá como guía para poder comenzar con el desarrollo de la aplicación, los requerimientos de mayor relevancia deberán poder ser soportados por la arquitectura especificada en este documento.

* 1. Definiciones, siglas y abreviaturas.
  2. Referencias

-Modelo de casos de uso

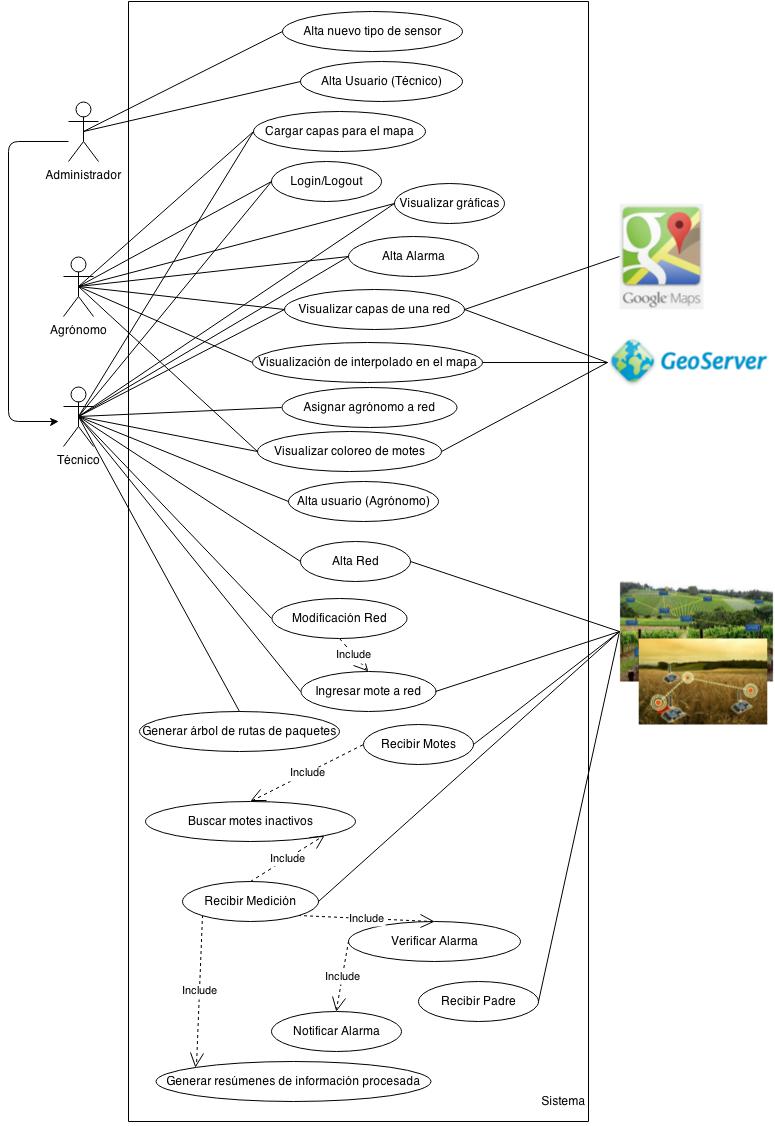
-Especificación de requerimientos

-Modelo de diseño

* 1. Visión general

Presenta primero una vista del modelo de casos de uso, conteniendo los casos de uso relevantes para la arquitectura; luego un diagrama de distribución que descompone el sistema en componentes.

1. Vista del Modelo de Casos de Uso
   1. Diagrama de Casos de Uso relevantes a la Arquitectura



* 1. Casos de Uso relevantes a la Arquitectura
     1. Alta nuevo tipo de sensor

Este caso de uso es ejecutado únicamente por el usuario administrador y sirve para que de alta en el sistema un nuevo tipo de sensor que podrá ser usado por todos los motes del sistema, la configuración entre otras cosas permite seleccionar colores y valores asociados a estos además debe definir el valor máximo y mínimo del mote, por lo que debe estar disponible al dar de alta un nuevo sensor en la base de datos, esto significa interacción en la base de datos y por tanto es de relevancia para la arquitectura.

* + 1. Login/Logout

El caso de uso consiste en un usuario de cualquier tipo que ingresa al sistema, para lo cual ingresara sus credenciales, el sistema lo valida e ingresa.

Posteriormente podrá realizar acciones en diferentes páginas por lo que la arquitectura deberá poder soportar el manejo de sesiones de usuario.

Logout por otro lado permite terminar la sesión iniciada en el CU Login.

* + 1. Alta alarma

Dar de alta una alarma en el sistema será de utilidad cuando un usuario quiera recibir notificaciones sobre mediciones extremas de los motes, por ejemplo temperaturas muy bajas que podrían dañar los cultivos.

Es de prioridad alta e implica interacción indirecta con el subsistema que se encargará de procesar los paquetes, revisar si la medición detectada levanta alguna alarma y finalmente notificar al usuario.

* + 1. Visualizar capas de una red

Este caso de uso permite al usuario ver una red de sensores sobre la que tiene permisos con una vista particular de capas que muestren información de interés para él, por ejemplo vista por alguna variable.

Al interactuar con el sistema externo Google Maps y GeoServer es de importancia para la arquitectura.

* + 1. Visualización de interpolado en el mapa

Permite a un usuario visualizar una variable interpolada en el mapa utilizando los datos capturados por la red que está visualizando.

Este caso de uso implica realizar varios cálculos y el uso de librerías externas además de ser de alta prioridad, por esto es relevante para la arquitectura.

* + 1. Alta Usuario (Agrónomo)

El caso de uso permite a un usuario de tipo administrador o técnico dar de alta un usuario de tipo agrónomo, se considera para la arquitectura por ser uno de los casos de uso base para el funcionamiento de otros casos de uso y no representa una complejidad muy alta.

* + 1. Alta Usuario (Técnico)

El caso de uso permite a un usuario de tipo administrador dar de alta un usuario de tipo técnico, se considera para la arquitectura por ser uno de los casos de uso base para el funcionamiento de otros casos de uso y no representa una complejidad muy alta.

* + 1. Alta red

Este caso es la base del sistema, y permite a un usuario de tipo administrador o técnico crear una nueva red en el sistema, para eso debe ingresar la dirección Ip del mote base de la red y su ubicación, a partir de esta dirección el sistema podrá descubrir al mote base, al descubrirlo lo agregara en la base de datos y comenzará a recibir información de los nuevos recursos que se instalen en la red, la instalación de la red continúa en el CU Modificar Red.

Este caso de uso es de alta prioridad y representa la base del funcionamiento del sistema por lo tanto es fundamental para la arquitectura.

* + 1. Modificar Red

Continúa con la instalación de una red, o modificación de una ya instalada, para lo cual permite ingresar o eliminar motes de la red, el ingreso de nuevos motes será la funcionalidad principal de este CU que desarrollaremos para la arquitectura.

* + 1. Ingresar mote a red

Este caso está incluido en el caso de uso Modificar Red y permite dar de alta un nuevo mote en una red, para lo cual el usuario selecciona de una lista de direcciones Ip, que fue generada a partir del reconocimiento del sistema de un mote ubicado físicamente en la red, la dirección del mote que quiere agregar y luego el sistema se suscribirá a todos los sensores de ese mote para que le envíe información constantemente. Dada la interacción entre el sistema externo de redes y el sistema es de alta prioridad para la arquitectura.

* + 1. Recibir Motes

El caso de uso lo inicia un mote de la red de sensores y le permite presentarse al sistema como un nuevo mote a través del mote base, para lo cual le envía al sistema su dirección IP, el sistema le pregunta qué sensores tiene y luego almacenará la información en la base de datos.

Esta interacción entre el sistema y la red de sensores es fundamental y por eso es de prioridad para la arquitectura.

* + 1. Asignar agrónomo a red

El caso de uso construye una relación entre un usuario agrónomo y una red de motes para que este tenga permisos para visualizar los datos que esta captura.

* + 1. Recibir Medición + Verificar Alarma +Notificar

El caso de uso se dispara cuando un mote, sobre el cual se está suscripto envía una medición, el sistema la captura la medición y dispara el caso de uso verificar alarma, que chequea si esta medición dispara alguna alarma, y en caso de que sea así se activa el caso de uso notificar que enviará una notificación a los usuarios interesados en la alarma.

* + 1. Recibir Padre

Cuando el sistema recibe un mensaje de un mote que indica que hubo un cambio de padre, la calidad del enlace y otros datos de interés el caso de uso se activa y almacena esa información que luego será utilizada para generar las rutas.

* + 1. Generar árbol de rutas de paquetes

El caso de uso permite a partir de los datos recibidos en el CU Recibir padre armar árboles con las rutas de paquetes, la arquitectura debe poder realizar este caso de uso con buena performance.

* + 1. Cargar capas para el mapa

Permite a un usuario de tipo agrónomo cargar archivos .shp para que puedan ser visualizados en el mapa, esto es de utilidad cuando se quiere representar en el mapa algún detalle del terreno que no está en los mapas de google.

* + 1. Visualizar coloreo de motes

Este caso de uso es fundamental y permite ver mediante colores sobre los motes que son configurados en el caso de uso Alta Nuevo Tipo de Sensor, los valores que estos capturan respecto de alguna variable, por ejemplo si capturan una temperatura de 30 grados o mayor veríamos un color rojo indicando calor.

* + 1. Visualizar gráficas

El caso de uso permite ver gráficas de las mediciones capturadas por los motes que seleccione el usuario, por ejemplo puede graficar varias variables de diferentes motes en función del tiempo, este caso de uso interactúa con aplicaciones de terceros para poder realizar las gráficas y por tanto es relevante para la arquitectura.

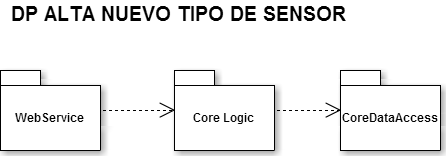
* + 1. Buscar motes inactivos

Este caso de uso permite detectar cuando un mote no envió mediciones luego de cierto tiempo o cuando la base no lo está reconociendo.

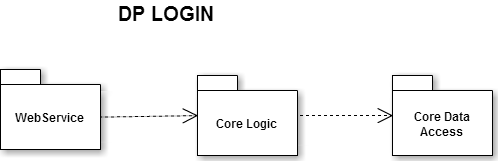
El caso de uso forma parte de la arquitectura ya que la misma debe soportar la verificación de inactividad de los motes.

1. Trazabilidad desde el Modelo de Casos de Uso al Modelo de Diseño
   1. Diagramas de paquetes

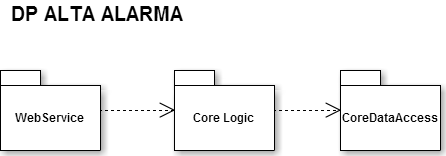
* + 1. Alta nuevo tipo de sensor



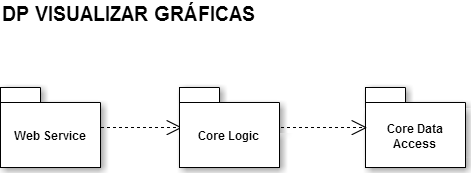
* + 1. Login/Logout



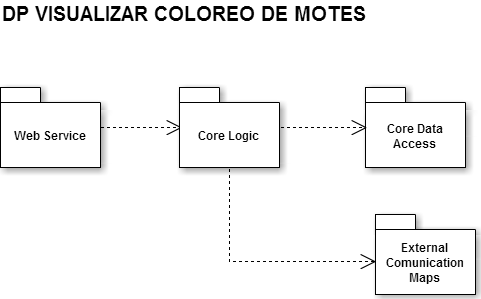
* + 1. Alta alarma.



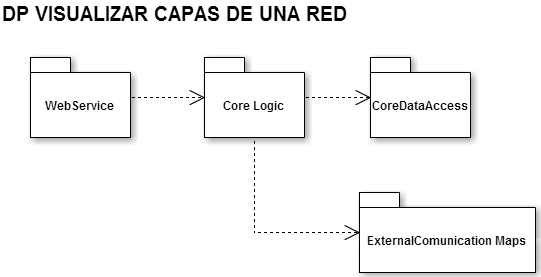
* + 1. Visualizar gráficas



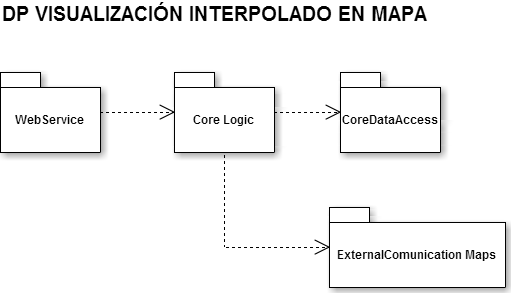
* + 1. Visualizar coloreo de motes



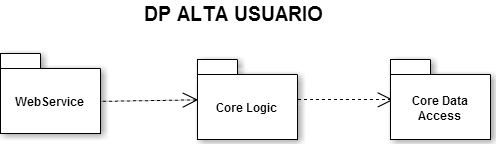
* + 1. Visualizar capas de una red



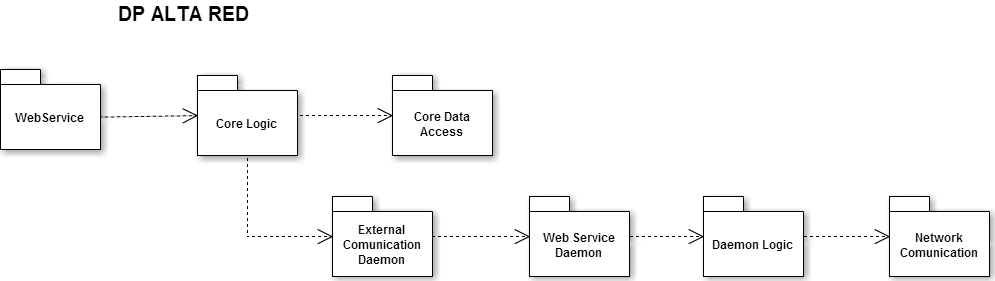
* + 1. Visualización de interpolado en el mapa



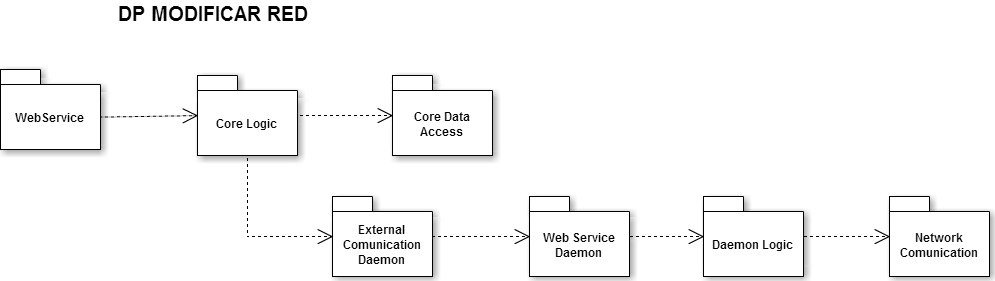
* + 1. Alta Usuario (Técnico/Agrónomo)



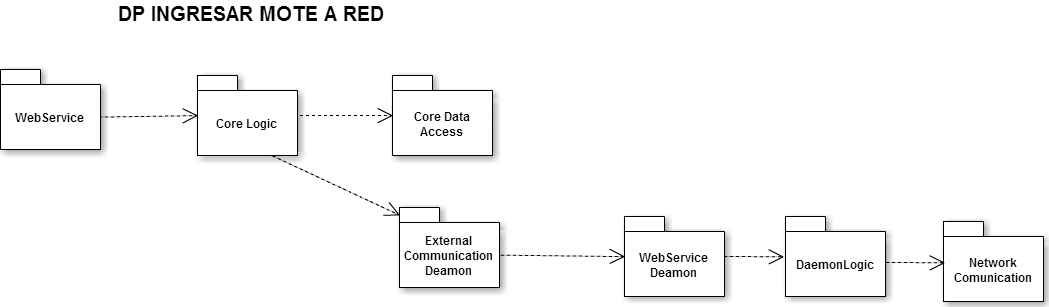
* + 1. Alta red



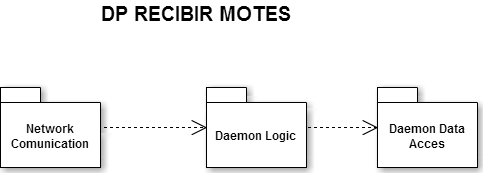
* + 1. Modificar Red



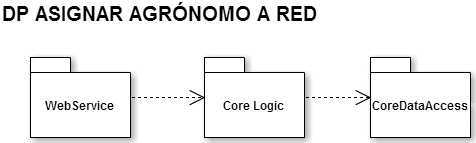
* + 1. Ingresar mote a red



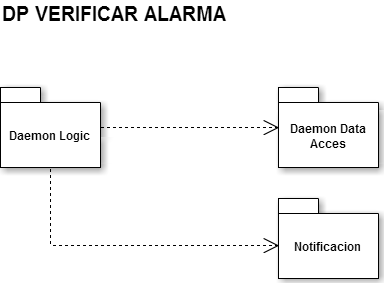
* + 1. Recibir Motes



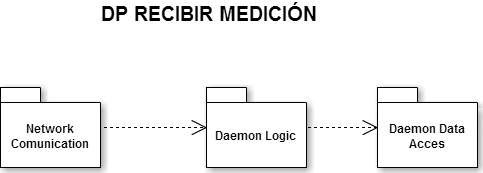
* + 1. Asignar agrónomo a red



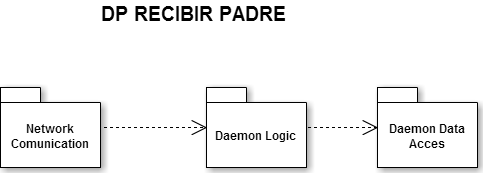
* + 1. Verificar Alarma



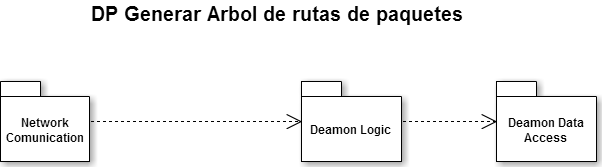
* + 1. Recibir Medición



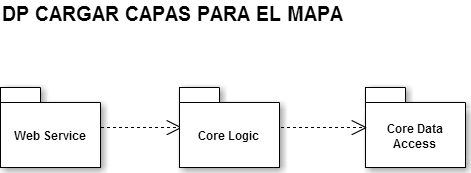
* + 1. Recibir Padre



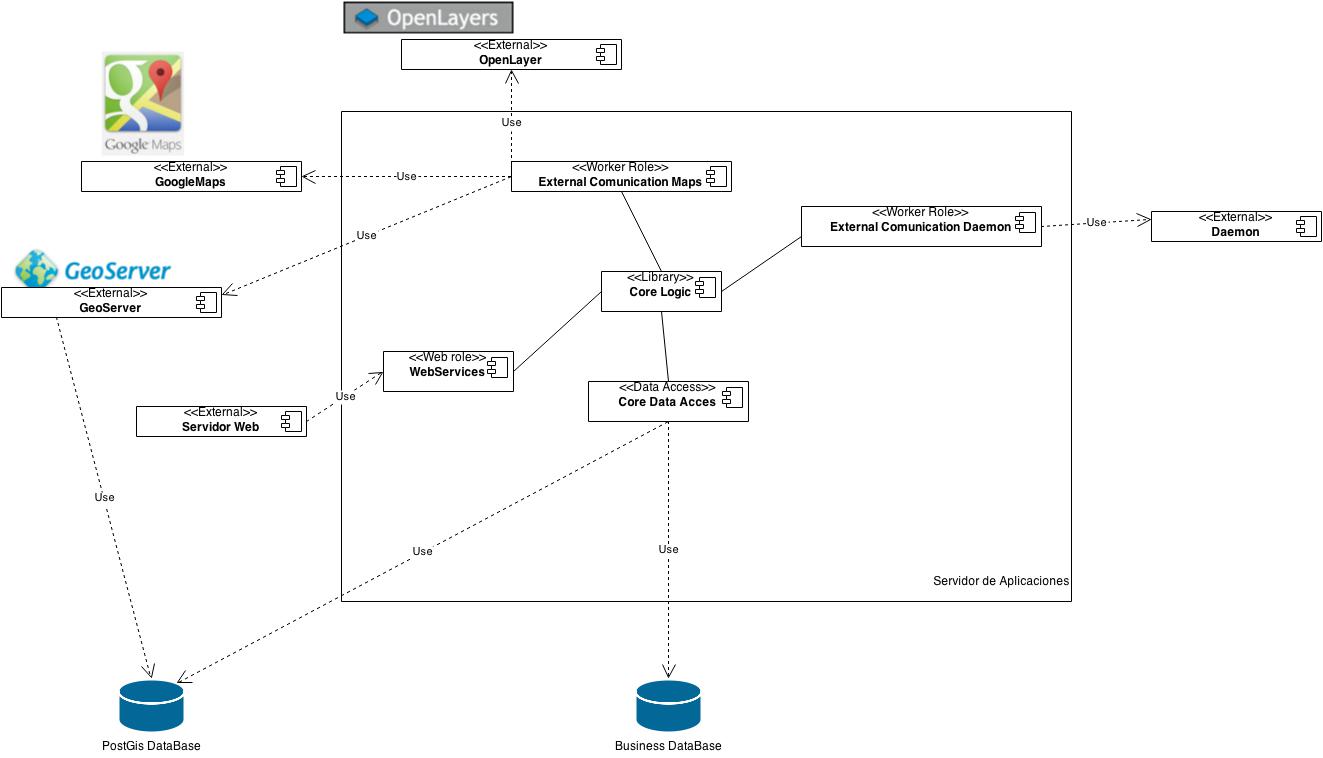
* + 1. Generar árbol de rutas de paquetes



* + 1. Cargar capas para el mapa



1. Vista del Modelo de Diseño
   1. Descomposición en subsistemas
      1. Servidor de Aplicaciones



* 1. Subsistemas específicos
     1. Core Logic

Componente encargado de la gestión de la lógica de negocio, incluye el manejo de usuarios, manejo de redes, procesamiento de algunos datos para que luego puedan ser mostrados, brindar servicios para la Web, etc.

* + 1. Core Data Access

Se encarga de realizar la gestión de la base de datos de negocio y la base de datos de Postgis orientada a los mapas, es importante separarlo de la lógica para prever el mínimo impacto ante cambios en la base de datos.

* + 1. ExternalComunicationMaps

Este componente gestionará las comunicaciones con entidades externas, ya que el sistema se debe adaptar a los cambios en ellas con el mínimo impacto.

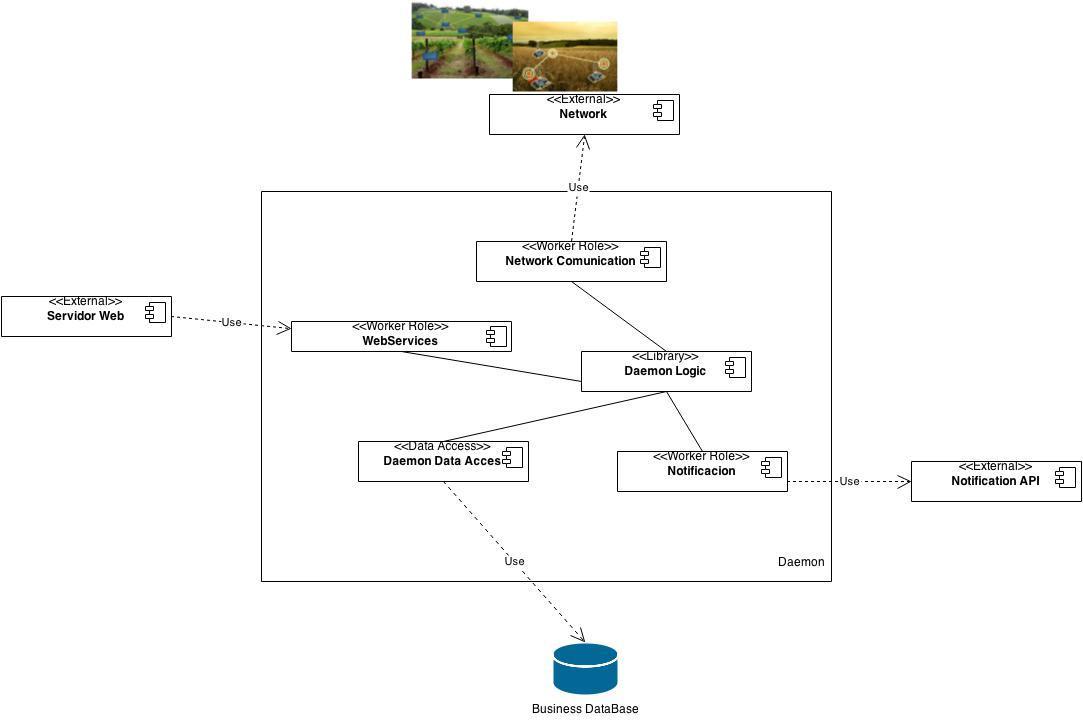
* + 1. ExternalComunication Network

Este componente es fundamental dentro del Servidor de Aplicaciones y gestiona las comunicaciones entre él y el subsistema Daemon, para lo cual el Daemon publicará servicios web que estarán disponibles para el Servidor de Aplicaciones, esto es necesario ya que ante modificaciones generadas por el usuario en la web, las modificaciones a nivel de suscripciones a los motes deben ser ágiles.

* + 1. Web Services

Componente que representa la lógica web que será accesible por el subsistema Servidor Web a través de Web Services.

* 1. Daemon



* 1. Subsistemas específicos
     1. Network Comunication

Componente separado de la lógica del subsistema Daemon que se encarga de gestionar la comunicación con la red de sensores, al estar está en fase de desarrollo tiene tendrá tendencia a cambiar en el corto plazo y por tanto el impacto de este cambio debe ser mínimo para la arquitectura.

* + 1. DaemonLogic

Este componente contiene la lógica del subsistema Daemon, y será básicamente un proceso de tipo “demonio” que esperara información de los motes de la red y realizará acciones según los datos que reciba.

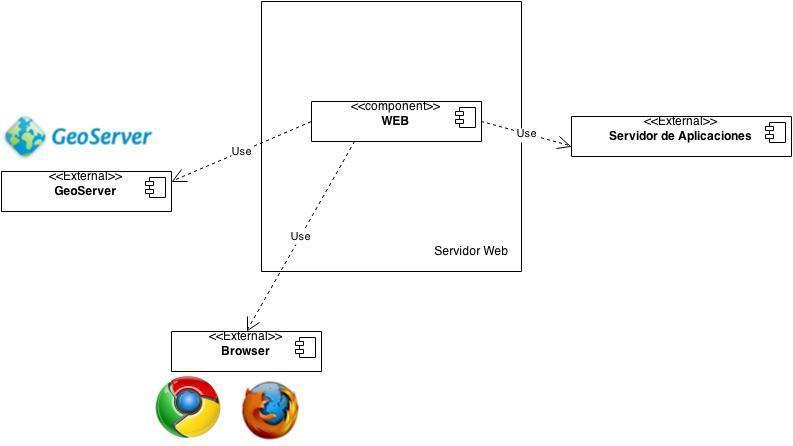
* + 1. Daemon Data Access

Accede a la base de datos para persistir los datos que le brinda el componente DaemonLogic, debe estar sincronizado de alguna manera con el componente Core Data Access del subsistema Servidor Web ya que la comunicación entre ellos es vía la base de datos de negocio.

* + 1. Web Services

Este componente contiene servicios web con operaciones que serán publicadas para el servidor de aplicaciones, estas son fundamentales para que el servidor de aplicaciones pueda manifestar acciones del usuario en la red de sensores.

* 1. Servidor Web

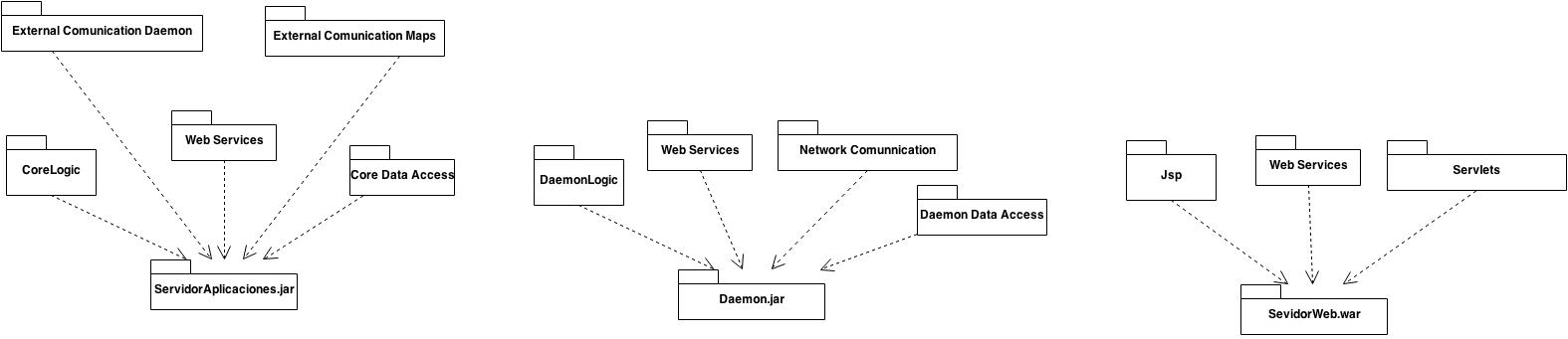


* 1. Subsistemas específicos
     1. Web

Este componente representa la lógica del servidor web y contiene básicamente las páginas .jsp y la lógica de las mismas que se presenta en los servlets.

1. Trazabilidad desde el Modelo de Diseño al Modelo de Implementación

El siguiente diagrama, especifica la trazabilidad entre los subsistemas y la implementación.



1. Vista del Modelo de Implementación
   1. Manejo de errores

Ante el pedido del cliente de que los errores sean descriptivos, los manejaremos de la siguiente manera:

Si el error es detectado en el sistema Servidor de Aplicaciones:

Si es un error manejable desde la lógica, generará el mensaje de error y lo enviará al Servidor Web para que lo despliegue debidamente directamente como un String, el cual indicara específicamente el punto de falla y una sugerencia de cómo recuperarse.

Si es una excepción, la enviará al Servidor Web para que este la capture y despliegue, indicando que ocurrió una excepción y el detalle de la misma.

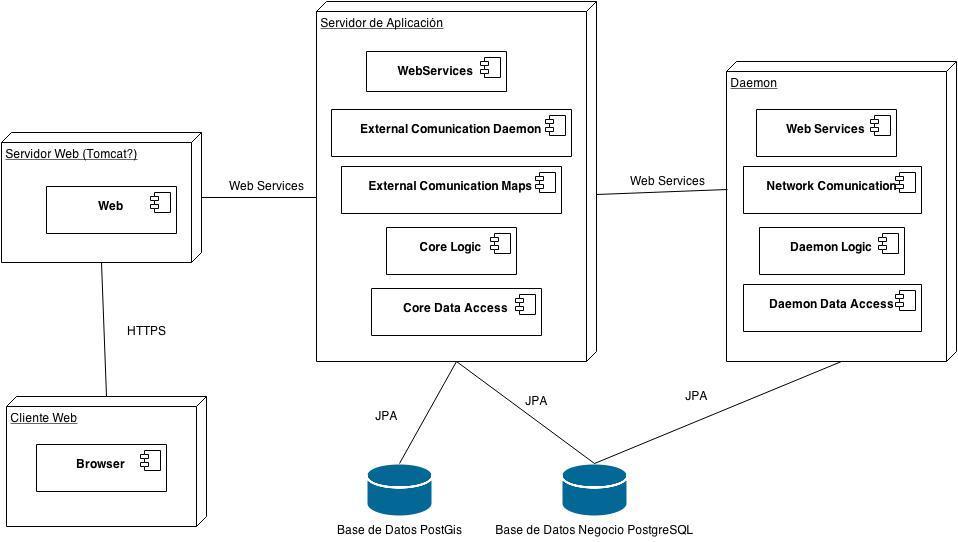
Si el error es detectado en el sistema Daemon aplicamos el mismo criterio que en el Servidor de Aplicaciones.

Si el error es en el Servidor Web, se manejará con el envío al usuario de mensajes descriptivos del error, este tipo de errores que ocurren pueden ser porque no completo campos obligatorios u otro tipo de errores del estilo, y el mensaje indicará claramente que debe hacer el usuario para revertir la situación.

También puede pasar que el error ocurrido en el Servidor Web no sea un error esperado, para lo cual el sistema desplegara un mensaje descriptivo para que el usuario sea notificado.

En ninguno de los casos la aplicación dejara de funcionar.

1. Vista del Modelo de Distribución
   1. Diagrama de Distribución



* 1. Nodos
     1. Cliente Web

El cliente web brinda la presentación del sistema para los usuarios finales, se encarga de desplegar la información que le brinda el servidor web a través del protocolo HTTPS en páginas web que corren en un navegador.

* + 1. Servidor de Aplicaciones

El servidor de aplicaciones es encargado de brindar el soporte a la aplicación de cliente web por lo que debe ser de alta disponibilidad.

Se encarga de mostrar la información que se encuentre en la base de datos al cliente, que se conectara al sitio mediante un navegador web, y le presenta la información en mapas o en graficas. Además se encarga de gestionar las suscripciones a los motes de la red de sensores, para que estos envíen información al sistema Daemon de forma automática.

Este componente puede correr en el mismo equipo que el Servidor de aplicaciones.

Debe interactuar con el componente externo GeoServer para el procesamiento de los datos geoespaciales.

* + 1. Daemon

El subsistema Daemon es un subsistema de tipo “demonio” que ejecuta continuamente recibiendo paquetes nuevos de las diferentes redes de sensores y también gestionando las configuraciones de red, tiene alta interacción con las redes y a medida que estas crezcan en tamaño el sistema procesara más. Debe correr de forma transparente para los usuarios del sistema, además puede instalarse en el mismo equipo que el Servidor de Aplicaciones.

El subsistema Daemon se corresponderá con el sistema externo Red de Sensores.

* + 1. Servidor Web

El servidor web es el encargado de crear la interfaz de páginas Web para que puedan ser consumidas por el Cliente Web en un navegador, para lo cual consume mediante Web Services la lógica del servidor de aplicaciones.

Este servidor corre en Apache Tomcat.

* + 1. Base de datos PostGIS

Base de datos encargada de dar persistencia a los datos geoespaciales, solo será accesible por el subsistema Servidor Web y también deberá tener interacción con el sistema externo GeoServer. Sin perder generalidad puede estar en el mismo lugar que la base de negocio pero será independiente.

* + 1. Base de datos PostgreSQL

Esta base de datos es accesible tanto por el subsistema Daemon como por el Servidor Web y se encarga de dar persistencia a la lógica de negocio, contiene entre otras cosas información de los usuarios, de gestión de la red, etc.

Sin perder generalidad puede estar en el mismo lugar que la base de mapas pero será independiente.

* 1. Conexiones
     1. Cliente Web – Servidor Web

El cliente se conecta con el servidor web a través del protocolo de capa de aplicación HTTP, y mediante mensajes HTTP POST/GET obtendrá la información del servidor para desplegarla en el navegador.

El enlace es a través de internet y debe ser de alta disponibilidad.

* + 1. Servidor de Aplicaciones – Base de Datos Postgis

La interconexión entre estos componentes se realiza mediante el API de java JPA que permite mapear tablas en la base de datos como clases que pueden ser manejadas por el sistema.

* + 1. Daemon – Base de Datos PostgreSQL

La interconexión entre estos componentes se realiza mediante el API de java JPA que permite mapear tablas en la base de datos como clases que pueden ser manejadas por el sistema.

* + 1. Servidor de Aplicaciones – Base de Datos PostgreSQL

La interconexión entre estos componentes se realiza mediante el API de java JPA que permite mapear tablas en la base de datos como clases que pueden ser manejadas por el sistema.

* + 1. Servidor de Aplicaciones – Daemon

La conexión entre estos subsistemas se realiza mediante web services que son publicados por el subsistema Daemon.

* + 1. Servidor Web – Servidor de Aplicaciones

Estos subsistemas se conectan mediante Web services que son publicados por el subsistema Servidor de aplicaciones.