

IoT: Medidor de Temperatura

Entrega 3

20/06/2016

**Universidad Tecnológica Nacional**

**Arquitectura de Software**

**IoT: Medidor de Temperatura**

Arquitectura de Software

Integrantes

* Appiano, Fabrizio 50497
* Mena, José Ignacio 60507
* Pattusi, Rodrigo 50590
* Ryser, Carlos 60420

Contenido

[1. Código fuente versionado en Github. 2](#_Toc453801472)

[2. Definición de Requerimientos y Supuestos 2](#_Toc453801473)

[Requerimientos Funcionales 2](#_Toc453801474)

[Supuestos 2](#_Toc453801475)

[Predicción de Evolución de la Arquitectura 2](#_Toc453801476)

[3. Definición de Requerimientos No Funcionales 2](#_Toc453801477)

[Requerimientos no funcionales 2](#_Toc453801478)

[4. Diagrama de Arquitectura 3](#_Toc453801479)

[5. Dependencias 6](#_Toc453801480)

[Configuración de Desarrollo 6](#_Toc453801481)

[Configuración de Testing 6](#_Toc453801482)

[Configuración de Producción 6](#_Toc453801483)

## Código fuente versionado en Github.

[Repositorio Github](https://github.com/rodrigogit45/ADS2016) (https://github.com/mappiano/ADS2016.git)

## Definición de Requerimientos y Supuestos

### Requerimientos Funcionales

* Tomar mediciones de temperatura ambiental.
* Informar de temperaturas máximas y mínimas alcanzadas.
* Los límites de temperaturas deben poder ser configurables.
* Se debe poder conocer la ubicación geográfica de donde proviene la medición de temperatura.

### Supuestos

* Se dará soporte a los siguientes web browser Chrome (versión 5.0 en adelante, Firefox (versión 3.2 en adelante), Edge (12.0 en adelante).
* Los navegadores antes mencionados tendrán activado el intérprete de JS.
* Todas las placas son de la misma marca y modelo.
* Cada lugar donde serán instaladas poseen conexión a internet mediante WiFi.
* Para poder realizar desarrollo y testing real (con los recursos disponibles) se asume un máximo de 3 placas Arduino conectadas simultáneamente al servidor central.

### Predicción de Evolución de la Arquitectura

* Estimamos que en futuras versiones se necesitará almacenar los datos de temperaturas tomadas, en un lugar centralizado o tener algún mecanismo de backup.
* La aplicación deberá considerar la posibilidad de que los navegadores no soportan JS y no permitan el uso de Cookies.
* Suponemos que un requerimiento futuro puede ser que el cliente desee poder buscar las mediciones de una placa determinada filtrando por área geográfica.
* El sistema deberá reconocer que una placa está siendo cambiada y el resultado final no debería alterar las mediciones ya tomadas.

## Definición de Requerimientos No Funcionales

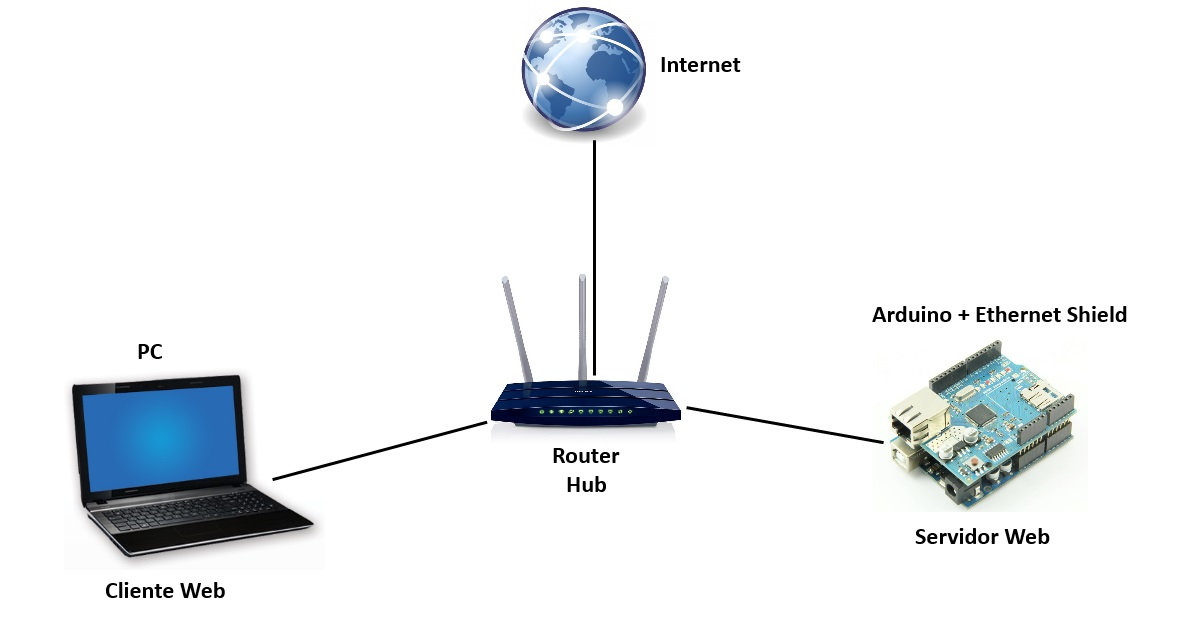
### Requerimientos no funcionales

1. La medición de la temperatura deberá ser realizada a través de una placa compatible con Arduino.
2. Las mediciones deberán ser visualizadas a lo largo del tiempo en un gráfico de barras.
3. El servidor web deberá estar dentro de la placa.
4. Las mediciones de temperaturas serán realizadas cada 5 segundos.
5. Debe brindarse una notificación sonora cuando la temperatura máxima o mínima sea marcada por el sensor de temperatura.
6. Todas las mediciones de temperatura deben dirigirse y almacenarse en el mismo servidor.
7. Debe haber una página distinta por cada placa para poder obtener las mediciones de cada placa Arduino.
8. Es necesario poder conocer el estado de las placas conectadas al sistema mediante un sistema “heartbeat”.

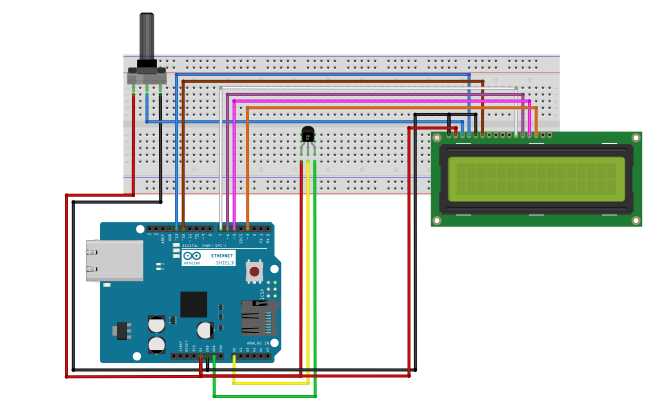
Para lograr el RNF 1 y RNF 2, la arquitectura poseerá una capa de interfaz de usuario independiente del resto del sistema que estará encargada de renderizar los gráficos necesarios para visualizar la información.

Para cumplir con el RNF 5 es necesario agregar una capa de persistencia de datos y considerar los servicios necesarios para la comunicación entre múltiples dispositivos Arduino y un único servidor (balanceo, concurrencia, acceso a la información, etc).

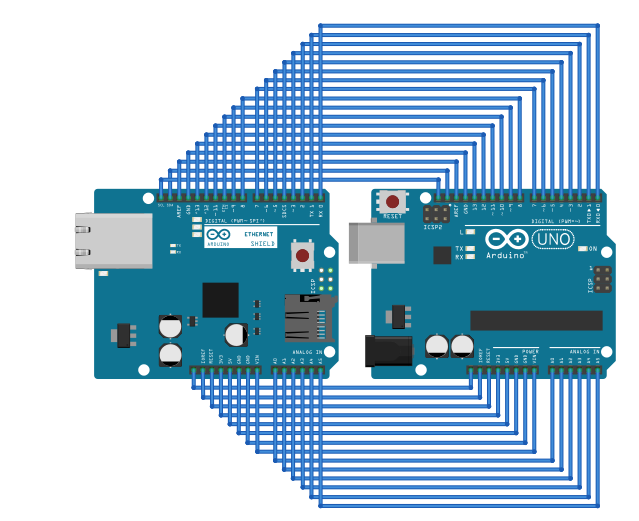
## Diagrama de Arquitectura



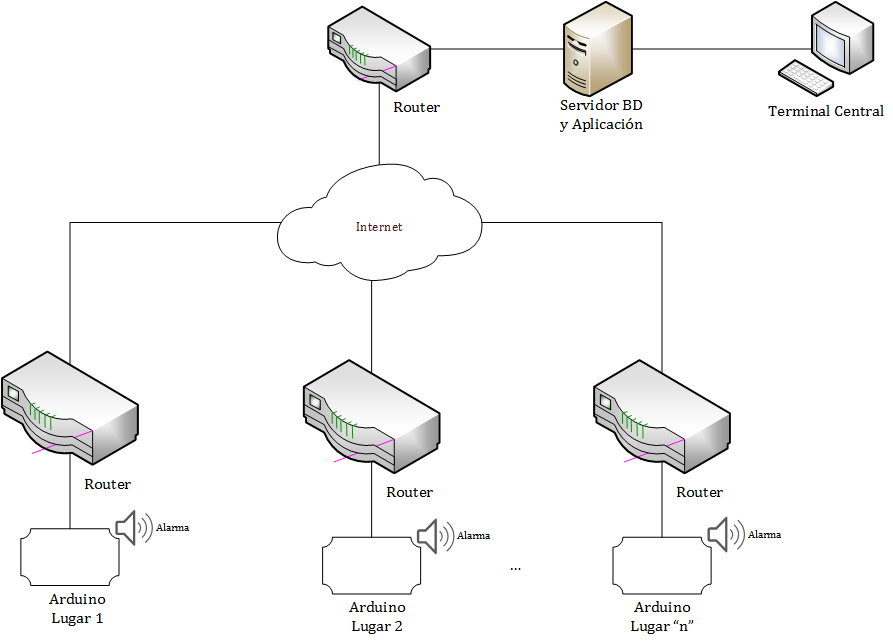
Plano del Circuito Ethernet con sensor de temperatura LM35 y Display de 16 pines.



*Conexión Arduino Uno R3 con Arduino Sheld Ethernet R3*



*Evolución de la arquitectura para cumplir con los RNF.*



Explicación

En la arquitectura que representa la evolución del sistema podemos observar que múltiples placas Arduino toman mediciones de temperatura desde diferentes ubicaciones y envían los datos obtenidos a un servidor central, a través de paquetes atravesando la red hasta llegar al servidor de BD donde estos son almacenados. Por otra parte, hemos considerado que el acceso a la información sea centralizada desde una sola terminal, la cual está conectada al servidor para así obtener la información y renderizarla en el gráfico de la página web. Esta arquitectura es completamente escalable ya que soporta todas las placas que se deseen con sólo añadir el hardware necesario (Arduinos, parlantes y routers) para soportar un aumento del procesamiento de información.

## Dependencias

### Configuración de Desarrollo

#### Lenguajes de programación utilizados

* **C:** este lenguaje nos permite trabajar sobre el código que interpreta la placa Arduino.
* **JavaScript:** es el lenguaje que interpretan los navegadores web y en este caso nos servirá para programar la interfaz del usuario.
* **HTML** **5:** no es un lenguaje propiamente dicho, será utilizado para diseñar la página web donde se mostrarán todas las muestras de temperatura.

#### Herramientas de desarrollo

* **Sublime:** es un editor de texto que permite la escritura de código de múltiples lenguajes. es muy sencillo de utilizar y tiene pocos requerimientos.

#### Librerías de terceros

* **Socket.io**
* **Express**

Entornos de ejecución

* **SQL Server 2014**: esta base de datos relacional nos permitirá almacenar los datos de las temperaturas tomadas por el sensor conectado a la placa Arduino.
* Cualquier navegador web que soporte las últimas actualizaciones de JS, HTML5 y   CSS3.
* Placa Arduino.
* El servidor de BD y de la aplicación será una PC con las siguientes características: Windows Server 2008, Procesador i7 6700 HQ, 8 GB de RAM, 1 TB de HD, conexión a internet con banda ancha de 15 Mb.

### Configuración de Testing

Igual a entorno de desarrollo.

### Configuración de Producción

Igual a entorno de desarrollo con entornos de ejecución

#### Entornos de Ejecución

* Cualquiera de los navegadores web que sean compatibles con las actualizaciones de HTML5, CSS3 y JavaScript.
* Motor de Base de Datos SQL Server 2014.
* La PC que hará de Servidor de BD y de host de la página web.

#### Plan de Seguimiento de Dependencias

Como no se poseen dependencias importantes fuera de la base de datos y las librerías JS se revisarán las dependencias cada  15 días.

#### Plan de Contingencia de Dependencias

* **Sublime**: En caso que nuestra licencia sea suspendida, el código se escribirá en otro editor como WebStorm por ejemplo, el cual podemos contar con licencia gratuita por ser estudiantes.
* **Sql Server 2014**: En caso de que nuestra licencia sea suspendida de manera imprevista, migraremos a MySql que es una base de datos Open Source.
* **Caída del servidor**: En caso de haber un corte de luz, se asistirá al servidor mediante un UPS. Si la caída es debido a una falla en el servidor o a la conexión a internet, el sistema en su totalidad dejará de ser funcional hasta solucionar dichos problemas.