



ESCUELA TÉCNICA SUPERIOR DE INGENIERÍAS INFORMÁTICA Y  
DE TELECOMUNICACIÓN

DEPARTAMENTO DE COMPUTACIÓN Y SISTEMAS INTELIGENTES

TRABAJO DE FIN DE GRADO:

**TIMESERIESANALYSIS: SISTEMA  
INTEGRAL PARA ANÁLISIS Y  
PREDICCIÓN DE SERIES TEMPORALES**

JAVIER MORENO VEGA

---

TUTOR DE PROYECTO:  
JOSÉ MANUEL BENÍTEZ SÁNCHEZ

<http://bahia.ugr.es/~jmoreno>

21 de junio de 2017

# Índice

<b>1. Introducción</b>	<b>2</b>
1.1. Motivación . . . . .	2
1.2. Objetivos . . . . .	3
1.3. Resumen del documento . . . . .	3
<b>2. Análisis y diseño</b>	<b>5</b>
2.1. Metodología de desarrollo . . . . .	5
2.2. Análisis de requisitos . . . . .	7
2.3. Diagramas . . . . .	24
2.4. Interfaces de usuario . . . . .	34
2.5. Herramientas utilizadas . . . . .	40
<b>3. Biblioteca de análisis (TimeSeriesAnalysisLibrary)</b>	<b>43</b>
3.1. Medidas de complejidad . . . . .	43
3.2. Transformaciones . . . . .	47
3.3. Clustering . . . . .	48
3.4. Predicción . . . . .	50
3.5. Clasificación . . . . .	51
<b>4. Implementación</b>	<b>52</b>
4.1. Despliegue en Cloud (DockersProject) . . . . .	52
4.2. Biblioteca de análisis en R (TimeSeriesAnalysisLibrary) . . . . .	54
4.3. API PHP (TimeSeriesAnalysisAPI) . . . . .	56
4.4. Angular aplicación web (TimeSeriesAnalysisApp) . . . . .	57
4.5. Base de datos MySQL . . . . .	58
<b>5. Conclusiones y vías futuras</b>	<b>59</b>
<b>6. Apéndices</b>	<b>62</b>
6.1. Apéndice A: Bibliografía . . . . .	62
6.2. Apéndice B: Glosario de Términos . . . . .	67

# 1. Introducción

## 1.1. Motivación

Las series temporales han sido, son y serán muy importantes. Resulta difícil imaginar una rama de las ciencias en la que no aparezcan datos que puedan ser considerados como series temporales.”[44]

Su definición podría ser una secuencias de datos medidos en determinados momentos y ordenados cronológicamente, pudiendo estar estos datos espaciados a intervalos iguales o desiguales. El principal uso de las series temporales es el de estudiar sus propiedades y obtener valores futuros.

Son un tipo de dato de enorme importancia en múltiples campos del conocimiento y actividad humanas. Su estudio ha preocupado a los científicos y técnicos desde hace mucho tiempo, razón por la cual existe una amplia literatura científica con propuestas de métodos para abordar su análisis (estudio de sus propiedades) y predicción (averiguar los valores futuros). Sin embargo, no existe ningún método universalmente válido pues las series pueden tener aspectos radicalmente distintos y no todos los métodos son válidos para todas. Además, en la actualidad no existe una plataforma con la que trabajar sobre series temporales sin tener que usar lenguajes de programación y eliminar la necesidad de conocimientos técnicos sobre implementación de esta funcionalidad.

Para disminuir la dificultad de un análisis de una serie temporal se ha pensado este proyecto haciendo uso de una aplicación web de un fácil uso. Desde esta aplicación web se podrá subir series temporales por los usuarios y poder aplicarles cálculos a la vez que tener una visión general.

Para analizar las series temporales haremos uso de métodos, que pueden ser transformaciones, métodos de predicción o clasificación y medidas de complejidad. Cada medida de complejidad se puede considerar como un cálculo que se hace sobre la serie temporal y devuelve un resultado numérico que identifica la complejidad de la misma. La complejidad es una medida del nivel de dificultad requerido para expresar o predecir las propiedades de un sistema [28]. En este proyecto se usa la complejidad de una serie para generar un árbol de clasificación y hacer experimentos e intentar generar una clasificación

de series temporales, consiguiendo posteriormente un método automático de clasificación de series temporales y elegir el método de predicción más adecuado.

## 1.2. Objetivos

El objetivo principal de este proyecto es construir un clasificador que nos asesore en la elección del método de predicción más adecuado. El conocimiento para este sistema se obtiene de analizar la base de datos de series temporales, construida para el proyecto. El análisis consiste en aplicar métodos para obtener unas medidas de cada serie temporal y poder compararlas. A estas medidas se las conoce como medidas de complejidad. Teniendo una lista de valores para cada serie se calculan grupos mediante clustering. Se obtiene la clasificación; y finalmente se calculan varios métodos de predicción sobre todas las series temporales y se selecciona de cada grupo el método con el menor error medio. La construcción de este sistema implica estudiar los principales métodos, adaptar las implementaciones disponibles o realizar implementaciones más efectivas, evaluarlos e integrarlos para que trabajen de forma cooperativa.

El objetivo secundario del proyecto es poner a disposición de los usuarios un conjunto de métodos de análisis y predicción de series temporales, que puedan ser aplicados de manera sencilla desde una aplicación web sin tener que entrar en demasiados detalles de parámetros de configuración o de la implementación de los mismos.

## 1.3. Resumen del documento

Esta memoria explica todo el trabajo desarrollado entrando en detalle en los módulos del sistema más importantes.

Primero se expone el análisis realizado para montar el sistema, atendiendo a: metodologías de desarrollo utilizadas, análisis de requisitos, diagramas, interfaces de usuario y las herramientas de terceros utilizadas.

Posteriormente se hace una explicación detallada de la biblioteca de análisis

que se ha desarrollado, explicando todos sus métodos, cada método explicado en su sección: medidas de complejidad, transformaciones, predicción y clasificación.

En tercer lugar se plantea la implementación que se ha seguido entrando en detalle en lenguajes usados y realizaciones técnicas.

Por último se muestran unas conclusiones, académicas y profesionales; incluyendo un resumen del trabajo desarrollado y los objetivos conseguidos.

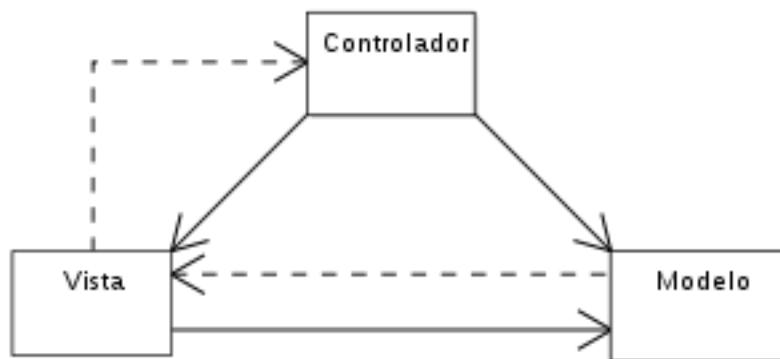
## 2. Análisis y diseño

En esta sección se explica el análisis que se ha realizado para la construcción de este proyecto junto con diagramas completos de los diversos módulos y capturas de pantalla de las interfaces de usuario.

### 2.1. Metodología de desarrollo

Durante el desarrollo se han usado diversas metodologías, paradigmas y patrones de arquitectura; cabe destacar como más importantes Scrum y Modelo-Vista-Controlador, esta última en la aplicación web, que usando Angular es muy sencillo implementarla.

El modelo-vista-controlador (mvc) [46] es un patrón de arquitectura de software, el cual divide el desarrollo de un sistema en tres módulos o partes principales, separando la interfaz de usuario (vista) de la lógica (controlador) y los datos (modelo). Este patrón beneficia en la separación de conceptos lo que facilita la tarea de desarrollo y mantenimiento, además de generar una gran abstracción entre los módulos. Se ha usado en la aplicación web con Angular, aunque también se podría considerar su uso en el sistema en general ya que la abstracción realizada en módulos encajaría con mvc; siendo la vista la aplicación web, el controlador la API y el modelo las dos bases de datos y la biblioteca de métodos de análisis. Por lo tanto tendríamos una arquitectura mvc (aplicación web) que a su vez pertenece a otra arquitectura mvc (sistema completo).



La metodología de desarrollo usada ha sido Scrum [50] que es una metodología de desarrollo ágil caracterizada por tener una estrategia de desarrollo incremental y una ejecución completa del producto por intervalos, equipos de

desarrollo auto organizados y solapamiento de las diferentes fases del desarrollo. Cada iteración del desarrollo es un "sprint" al finalizar cada uno se crea un incremento de software utilizable, la duración de estos suele ser entre 2 y 3 semanas.

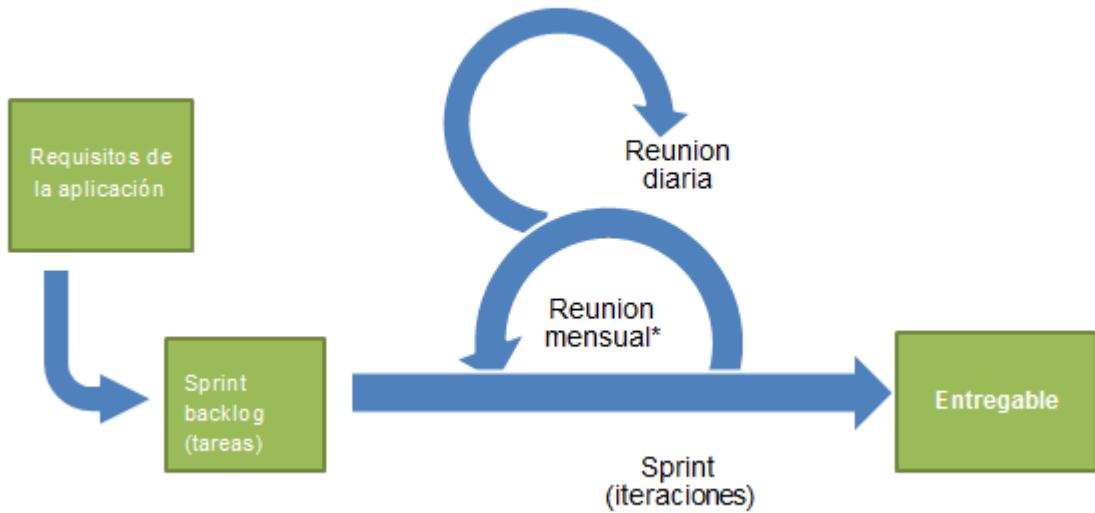
Los roles principales en scrum son: Product owner (representa la voz del cliente), scrum master (elimina obstáculos dentro del equipo y se encarga de que funcione scrum) y el equipo de desarrollo (puede estar formado por desarrolladores de software, testers, documentadores, etc). Cada equipo debe tener estos roles bien definidos.

También se realizan diversas reuniones: Daily scrum (reuniones diarias para ver como va el desarrollo), planificación (como su propio nombre indica planificar el desarrollo que se realizará en los próximos sprints), revisión del sprint (presentar el trabajo desarrollado) y retrospectiva del sprint (los miembros del equipo proponen mejorar y problemas que han podido ocurrir durante el sprint).

Todo el desarrollo que se va a realizar se agrupa en documentos llamados historias de usuario, cada historia de usuario contiene una funcionalidad completa y bien definida, y estas se dividen a su vez en tareas de desarrollo. Y todas las historias el equipo las tiene que puntuar con puntos de historia que deben seguir la sucesión de Fibonacci [52] (0, 1, 1, 2, 3, 5, 8, 13, ...), esto es indicativo para ver la dificultad de cada historia, a mayor dificultad mayor puntuación.

Los elementos usados para el desarrollo son: product backlog (agrupación de todas las historias de usuario del proyecto), sprint backlog (subconjunto de historias de usuario que se realizarán, o se intentarán, en el sprint), burn down chart (gráfica que muestra el progreso del desarrollo conforme se van acabando historias de usuario y reduciendo el número de puntos de historia restantes).

La elección de esta metodología es debido a tener experiencia propia por haber trabajado profesionalmente con ella, pero sobre todo el que ofrezca un desarrollo iterativo e incremental evitando el seguimiento de un estricto plan, adaptándose de forma continua a las circunstancias del proyecto.



## 2.2. Análisis de requisitos

En esta sección se realiza el análisis de requisitos del sistema y como se hace uso de Scrum los casos de uso se estructuran en historias de usuario y se incluyen en el backlog.

El desarrollo se ha realizado en dos fases con un periodo de 1 mes entre ambas donde se realizó investigación.

Después de un tiempo de estudio de los requisitos y la funcionalidad que debía tener el sistema se han obtenido las siguientes historias de usuario que formaran el product backlog en la primera fase de desarrollo.

Consideraciones:

- **Usuario anónimo:** Es considerado como un usuario que no ha iniciado sesión en la aplicación.
- **Usuario registrado:** Es considerado como un usuario que ha iniciado sesión correctamente en la aplicación.
- **Usuario:** Es considerado como un usuario general pudiendo haber iniciado sesión o no.

Crear una nueva cuenta de usuario			
<b>Identificador:</b> HU001	<b>Prioridad:</b> 1	<b>Iteración:</b> 1	<b>Puntos de historia:</b> 5
<b>Descripción:</b> Como usuario anónimo quiero poder crear una nueva cuenta.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Construir la capa de datos en el servidor para crear usuarios (base de datos SQL).</li> <li>· Construir método en la API para crear nuevas cuentas de usuario.</li> <li>· Construir método en la API para validar datos.</li> <li>· Construir la interfaz de usuario para crear la cuenta.</li> <li>· Implementar conexiones con la API desde el cliente.</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Crear cuenta de usuario correctamente.</li> <li>· Validar datos correctamente.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>			
<b>Observaciones:</b>			

Iniciar sesión			
<b>Identificador:</b> HU002	<b>Prioridad:</b> 1	<b>Iteración:</b> 2	<b>Puntos de historia:</b> 5
<b>Descripción:</b> Como usuario anónimo quiero poder iniciar sesión.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Construir método en la API para iniciar sesión como usuario.</li> <li>· Construir la interfaz de usuario para iniciar sesión.</li> <li>· Implementar conexiones con la API desde el cliente.</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Iniciar sesión correctamente.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>			
<b>Observaciones:</b> Depende de HU001 para poder crear la cuenta de usuario antes de iniciar sesión.			

Subir serie temporal			
<b>Identificador:</b> HU003	<b>Prioridad:</b> 1	<b>Iteración:</b> 2	<b>Puntos de historia:</b> 8
<b>Descripción:</b> Como usuario registrado quiero poder subir/crear una nueva serie temporal pública o privada.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Construir la capa de datos en el servidor para crear series temporales (base de datos SQL).</li> <li>· Construir la capa de datos en el servidor para crear series temporales (base de datos InfluxDB).</li> <li>· Añadir método a la biblioteca de métodos R para calcular propiedades de una serie temporal</li> <li>· Construir método en la API para crear series temporales llamando al método de la biblioteca R para calcular propiedades de la serie temporal.</li> <li>· Construir la interfaz de usuario para subir series temporales.</li> <li>· Construir método en la API para validar datos.</li> <li>· Implementar conexiones con la API desde el cliente.</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Subir serie temporal correctamente.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>			
<b>Observaciones:</b> Depende de HU001 para poder crear las series temporales habiendo iniciado sesión.			

Listado de series temporales públicas			
<b>Identificador:</b> HU004	<b>Prioridad:</b> 1	<b>Iteración:</b> 3	<b>Puntos de historia:</b> 2
<b>Descripción:</b> Como usuario quiero poder ver un listado de las series temporales públicas almacenadas.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Construir método en la API para devolver un listado de series temporales públicas almacenadas.</li> <li>· Construir la interfaz de usuario para ver listado de series temporales.</li> <li>· Implementar conexiones con la API desde el cliente.</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Listar series temporales públicas almacenadas.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>			
<b>Observaciones:</b> Depende de HU003 para tener series temporales almacenadas.			

Listado de series temporales de usuario			
<b>Identificador:</b> HU005	<b>Prioridad:</b> 2	<b>Iteración:</b> 3	<b>Puntos de historia:</b> 3
<b>Descripción:</b> Como usuario registrado quiero poder ver un listado de mis series temporales.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Construir método en la API para devolver un listado de series temporales de un usuario almacenadas.</li> <li>· Construir la interfaz de usuario para ver listado de series temporales propias almacenadas.</li> <li>· Implementar conexiones con la API desde el cliente.</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Listar series temporales de un usuario almacenadas.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>			
<b>Observaciones:</b> Depende de HU003 para tener series temporales almacenadas.			

Mostrar serie temporal			
<b>Identificador:</b> HU006	<b>Prioridad:</b> 1	<b>Iteración:</b> 3	<b>Puntos de historia:</b> 3
<b>Descripción:</b> Como usuario quiero poder ver una serie temporal.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Construir método en la API para devolver una serie temporal sin valores.</li> <li>· Construir la interfaz de usuario para ver una serie temporal</li> <li>· Implementar conexiones con la API desde el cliente.</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Mostrar serie temporal.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>			
<b>Observaciones:</b> Depende de HU003 para tener series temporales almacenadas.			

<b>Mostrar gráfica de serie temporal</b>			
<b>Identificador:</b> HU007	<b>Prioridad:</b> 1	<b>Iteración:</b> 4	<b>Puntos de historia:</b> 3
<b>Descripción:</b> Como usuario quiero poder ver una gráfica de la serie temporal.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Modificar método de la API para devolver una serie temporal con valores.</li> <li>· Construir la interfaz de usuario para ver una gráfica de la serie temporal</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Mostrar gráfica de la serie temporal.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>			
<b>Observaciones:</b> Depende de HU006 para poder acceder a la serie temporal.			

<b>Mostrar características principales de una serie temporal</b>			
<b>Identificador:</b> HU008	<b>Prioridad:</b> 1	<b>Iteración:</b> 4	<b>Puntos de historia:</b> 2
<b>Descripción:</b> Como usuario quiero poder ver las características principales definidas y obtenidas de una serie temporal.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Modificar la interfaz de usuario para ver nuevas propiedades de una serie temporal</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Mostrar propiedades de la serie temporal.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>			
<b>Observaciones:</b> Depende de HU006 para poder acceder a la serie temporal.			

<b>Mostrar métodos de análisis de una serie temporal</b>						
<b>Identificador:</b> HU009   <b>Prioridad:</b> 1   <b>Iteración:</b> 4   <b>Puntos de historia:</b> 3						
<p><b>Descripción:</b>            Como usuario quiero poder ver un listado de selección con los tipos de análisis de series temporales disponibles.</p>						
<p><b>Tareas:</b></p> <ul style="list-style-type: none"> <li>· Añadir funcionalidad a la biblioteca de métodos R para obtener los métodos de análisis.</li> <li>· Añadir funcionalidad a la API para conectar con la biblioteca de métodos R.</li> <li>· Modificar la interfaz de usuario para mostrar un selector con los tipos de métodos de análisis disponibles.</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>						
<p><b>Pruebas de aceptación:</b></p> <ul style="list-style-type: none"> <li>· Mostrar selector con los diferentes métodos de análisis.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>						
<p><b>Observaciones:</b>            Depende de HU006 para poder acceder a la interfaz de la serie temporal.</p>						
<b>Mostrar selector de tipo de serie temporal a la que aplicar método</b>						
<b>Identificador:</b> HU010   <b>Prioridad:</b> 1   <b>Iteración:</b> 5   <b>Puntos de historia:</b> 3						
<p><b>Descripción:</b>            Como usuario quiero poder ver un listado de las series temporales a las que aplicar los métodos de análisis</p>						
<p><b>Tareas:</b></p> <ul style="list-style-type: none"> <li>· Añadir funcionalidad a la biblioteca de métodos R para obtener un listado de métodos de transformación.</li> <li>· Añadir funcionalidad a la API para conectar con la biblioteca de métodos R.</li> <li>· Modificar la interfaz de usuario para mostrar un selector con los tipos de series temporales transformadas y la original.</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>						
<p><b>Pruebas de aceptación:</b></p> <ul style="list-style-type: none"> <li>· Mostrar selector con las diferentes series temporales.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>						
<p><b>Observaciones:</b>            Depende de HU009 para añadir el selector del tipo de serie temporal.</p>						

<b>Aplicar métodos de transformación a una serie temporal</b>			
<b>Identificador:</b> HU011	<b>Prioridad:</b> 1	<b>Iteración:</b> 6	<b>Puntos de historia:</b> 5
<b>Descripción:</b> Como usuario quiero poder aplicar transformaciones a una serie temporal original o transformada y poder ver una gráfica con la serie transformada.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Añadir funcionalidad a la biblioteca de métodos R para aplicar métodos de transformación a una serie temporal.</li> <li>· Añadir funcionalidad a la API para conectar con la biblioteca de métodos R.</li> <li>· Modificar la interfaz de usuario para mostrar un selector con los diferentes métodos de transformación.</li> <li>· Modificar la interfaz de usuario para aplicar métodos de transformación a una serie temporal.</li> <li>· Modificar la interfaz de usuario para mostrar la gráfica de la transformación de una serie temporal.</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Mostrar gráfica de transformación de la serie temporal.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>			
<b>Observaciones:</b> Depende de HU010 para aplicar métodos a la serie temporal.			

<b>Aplicar medidas de complejidad a una serie temporal</b>			
<b>Identificador:</b> HU012   <b>Prioridad:</b> 1   <b>Iteración:</b> 6   <b>Puntos de historia:</b> 5			
<b>Descripción:</b> Como usuario quiero poder aplicar medidas de complejidad a una serie temporal original o transformada y poder ver los resultados.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Añadir funcionalidad a la biblioteca de métodos R para obtener un listado de medidas de complejidad.</li> <li>· Añadir funcionalidad a la biblioteca de métodos R para aplicar medidas de complejidad a una serie temporal</li> <li>· Añadir funcionalidad a la API para conectar con la biblioteca de métodos R.</li> <li>· Modificar la interfaz de usuario para mostrar un selector con las diferentes medidas de complejidad.</li> <li>· Modificar la interfaz de usuario para aplicar medidas de complejidad a una serie temporal</li> <li>· Modificar la interfaz de usuario para mostrar los resultados</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Mostrar resultados de complejidad de la serie temporal.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>			
<b>Observaciones:</b> Depende de HU010 para aplicar métodos a la serie temporal.			

Aplicar métodos de predicción a una serie temporal			
Identificador:	HU013	Prioridad:	1
Iteración:	6	Puntos de historia:	5
<b>Descripción:</b> Como usuario quiero poder aplicar métodos de predicción a una serie temporal original o transformada y poder ver una gráfica con la predicción y demás resultados.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Añadir funcionalidad a la biblioteca de métodos R para obtener un listado de métodos de predicción.</li> <li>· Añadir funcionalidad a la biblioteca de métodos R para aplicar métodos de predicción a una serie temporal</li> <li>· Añadir funcionalidad a la API para conectar con la biblioteca de métodos R.</li> <li>· Modificar la interfaz de usuario para mostrar un selector con los diferentes métodos de predicción.</li> <li>· Modificar la interfaz de usuario para aplicar métodos de predicción a una serie temporal</li> <li>· Modificar la interfaz de usuario para mostrar los resultados añadiendo una gráfica con los valores futuros.</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Mostrar resultados de predicción de la serie temporal incluyendo una gráfica con los valores futuros.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>			
<b>Observaciones:</b> Depende de HU010 para aplicar métodos a la serie temporal.			

<b>Aplicar métodos de clasificación a una serie temporal</b>			
<b>Identificador:</b> HU014	<b>Prioridad:</b> 1	<b>Iteración:</b> 6	<b>Puntos de historia:</b> 5
<b>Descripción:</b> Como usuario quiero poder aplicar métodos de clasificación a una serie temporal original o transformada y poder ver los resultados.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Añadir funcionalidad a la biblioteca de métodos R para obtener un listado de métodos de clasificación.</li> <li>· Añadir funcionalidad a la biblioteca de métodos R para aplicar métodos de clasificación a una serie temporal</li> <li>· Añadir funcionalidad a la API para conectar con la biblioteca de métodos R.</li> <li>· Modificar la interfaz de usuario para mostrar un selector con los diferentes métodos de clasificación.</li> <li>· Modificar la interfaz de usuario para aplicar métodos de clasificación a una serie temporal</li> <li>· Modificar la interfaz de usuario para mostrar los resultados.</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Mostrar resultados de clasificación de la serie temporal.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>			
<b>Observaciones:</b> Depende de HU010 para aplicar métodos a la serie temporal.			

<b>Mostrar series temporales privadas a sus usuarios</b>			
<b>Identificador:</b> HU015	<b>Prioridad:</b> 2	<b>Iteración:</b> 3	<b>Puntos de historia:</b> 3
<b>Descripción:</b> Como usuario registrado quiero poder acceder a mis series temporales privadas.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Añadir autenticación en la API al intentar acceder a series temporales privadas.</li> <li>· Añadir mensaje de error en la interfaz al acceder sin permisos a una serie temporal privada.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Mostrar serie temporal privada si es del usuario.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>			
<b>Observaciones:</b> Depende de HU003 para tener series temporales almacenadas.			

Acceder a perfil de usuario registrado			
<b>Identificador:</b> HU016	<b>Prioridad:</b> 3	<b>Iteración:</b> 3	<b>Puntos de historia:</b> 3
<b>Descripción:</b> Como usuario registrado quiero poder acceder a mi perfil.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Añadir menú de usuario a la pantalla de análisis</li> <li>· Añadir método de obtener usuario en la API.</li> <li>· Construir interfaz de edición de perfil de usuario registrado.</li> <li>· Añadir a la interfaz el listado de las series temporales del usuario.</li> <li>· Implementar conexiones con la API desde el cliente.</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Mostrar correctamente perfil de usuario registrado.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>			
<b>Observaciones:</b> Depende de HU001 para poder iniciar sesión.			

Editar perfil de usuario registrado			
<b>Identificador:</b> HU017	<b>Prioridad:</b> 3	<b>Iteración:</b> 3	<b>Puntos de historia:</b> 3
<b>Descripción:</b> Como usuario registrado quiero poder editar mi perfil.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Añadir método con autenticación en la API para poder modificar datos de usuario registrado</li> <li>· Añadir método de validación de datos en la API.</li> <li>· Construir interfaz de edición de perfil de usuario registrado.</li> <li>· Implementar conexiones con la API desde el cliente.</li> <li>· Gestionar las diferentes respuestas en el cliente.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Modificar correctamente perfil de usuario registrado.</li> <li>· Mostrar diferentes errores en el cliente.</li> </ul>			
<b>Observaciones:</b> Depende de HU001 para poder iniciar sesión.			

Una vez se han obtenido las historias de usuario es necesario hacer una aproximación de la velocidad de desarrollo para hacer las divisiones de Sprint.

La duración de un Sprint en el proyecto será de dos semanas, por tanto se tienen 15 días de desarrollo por Sprint. Y tendremos como velocidad estimada por Sprint de 8 puntos de historia.

Considerando que se tienen 63 puntos de historia del proyecto completo y la velocidad de desarrollo se obtienen un total de 8 Sprint.

A continuación se muestran las historias de usuario divididas por Sprint, para la primera fase, desarrollándose primero las que más prioridad tienen. Con la finalización de estos primeros 8 Sprint obtendremos la versión Alpha-2.0.0 del producto.

<b>Sprint: 1</b>		<b>Inicio:</b> 25/07/2016
Versión	Alpha-2.0.0	
Identificador	Historia de usuario	
HU001	Crear una nueva cuenta de usuario	
HU002	Iniciar sesión	

<b>Sprint: 2</b>		<b>Inicio:</b> 08/08/2016
Versión	Alpha-2.0.0	
Identificador	Historia de usuario	
HU003	Subir serie temporal	

<b>Sprint: 3</b>		<b>Inicio:</b> 22/08/2016
Versión	Alpha-2.0.0	
Identificador	Historia de usuario	
HU004	Listado de series temporales públicas	
HU006	Mostrar serie temporal	
HU007	Mostrar gráfica de serie temporal	

<b>Sprint: 4</b>		<b>Inicio:</b> 05/09/2016
Versión	Alpha-2.0.0	
Identificador	Historia de usuario	
HU008	Mostrar características principales de una serie temporal	
HU009	Mostrar métodos de análisis de una serie temporal	
HU010	Mostrar selector de tipo de serie temporal a la que aplicar método	

<b>Sprint:</b> 5	<b>Inicio:</b> 19/09/2016
Versión	Alpha-2.0.0
Identificador	Historia de usuario
HU005	Listado de series temporales de usuario
HU011	Aplicar métodos de transformación a una serie temporal

<b>Sprint:</b> 6	<b>Inicio:</b> 03/10/2016
Versión	Alpha-2.0.0
Identificador	Historia de usuario
HU012	Aplicar medidas de complejidad a una serie temporal
HU015	Mostrar series temporales privadas a sus usuarios

<b>Sprint:</b> 7	<b>Inicio:</b> 17/10/2016
Versión	Alpha-2.0.0
Identificador	Historia de usuario
HU013	Aplicar métodos de predicción a una serie temporal
HU016	Acceder a perfil de usuario registrado

<b>Sprint:</b> 8	<b>Inicio:</b> 31/10/2016
Versión	Alpha-2.0.0
Identificador	Historia de usuario
HU014	Aplicar métodos de clasificación a una serie temporal
HU017	Editar perfil de un usuario registrado

La segunda fase del desarrollo comienza un mes después de la finalización de la primera. A continuación se muestran las historias obtenidas para esta fase.

Misceláneos 1			
<b>Identificador:</b> HU018	<b>Prioridad:</b> 1	<b>Iteración:</b> 1	<b>Puntos de historia:</b> 5
<b>Descripción:</b> Historia de usuario para refactorizar y dividir paquete R TimeSerie en varios paquetes más pequeños.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Crear paquete R DataUtils</li> <li>· Crear paquete R TimeSeries</li> <li>· Crear paquete R TimeSeriesAnalysis</li> <li>· Crear paquete R Clustering</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Tests funcionando en paquete DataUtils.</li> <li>· Tests funcionando en paquete TimeSeries.</li> <li>· Tests funcionando en paquete TimeSeriesAnalysis.</li> <li>· Tests funcionando en paquete Clustering.</li> <li>· TimeSeriesAnalysisWebApi sigue funcionando correctamente.</li> </ul>			
<b>Observaciones:</b>			

Misceláneos 2			
<b>Identificador:</b> HU019	<b>Prioridad:</b> 3	<b>Iteración:</b> 1	<b>Puntos de historia:</b> 3
<b>Descripción:</b> Historia de usuario para refactorización y mejoras.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Mejorar automatización de Dockers.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Mantener funcionamiento correcto de base de datos de series temporales.</li> <li>· Mantener funcionamiento correcto de script de gestión de Docker.</li> </ul>			
<b>Observaciones:</b>			

Misceláneos 3			
<b>Identificador:</b> HU020	<b>Prioridad:</b> 2	<b>Iteración:</b> 1	<b>Puntos de historia:</b> 3
<b>Descripción:</b> Historia de usuario para refactorizar las URIs de TimeSeriesAnalysisWebApi.			
<b>Tareas:</b> <ul style="list-style-type: none"> <li>· Mejorar URIs de TimeSeriesAnalysisWebApi.</li> <li>· Mejorar API y URIS WebApp.</li> </ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"> <li>· Mantener funcionamiento correcto de la API.</li> </ul>			
<b>Observaciones:</b>			

<b>Misceláneos 4</b>
<b>Identificador:</b> HU021   <b>Prioridad:</b> 2   <b>Iteración:</b> 1   <b>Puntos de historia:</b> 3
<b>Descripción:</b> Historia de usuario para refactorización y mejoras.
<b>Tareas:</b> · Añadir PHP tests
<b>Pruebas de aceptación:</b> · Mantener funcionamiento correcto de la API. · Mantener funcionamiento correcto de la App.
<b>Observaciones:</b>

<b>Clasificar series temporales (Mejoras)</b>
<b>Identificador:</b> HU022   <b>Prioridad:</b> 1   <b>Iteración:</b> 1   <b>Puntos de historia:</b> 8
<b>Descripción:</b> Como usuario quiero poder clasificar series temporales correctamente según una clasificación.
<b>Tareas:</b> · Añadir InfluxDB Docker. · Mejorar lectura de TimeSeries R (Date) / Leer desde DB TimeSeries R · Scripts de investigación
<b>Pruebas de aceptación:</b> · Clasificar correctamente una serie temporal.
<b>Observaciones:</b>

<b>Documentación General (DOC)</b>
<b>Identificador:</b> HU023   <b>Prioridad:</b> 3   <b>Iteración:</b> 5   <b>Puntos de historia:</b> 3
<b>Descripción:</b> Historia de usuario para añadir la documentación restante a la memoria.
<b>Tareas:</b> · Añadir youtrack y teamcity al software · Explicar gestión de Docker's. · Actualizar diagramas. · Conclusiones académicas
<b>Pruebas de aceptación:</b> ·
<b>Observaciones:</b>

Clasificación de serie temporal (DOC)			
<b>Identificador:</b> HU024	<b>Prioridad:</b> 1	<b>Iteración:</b> 2	<b>Puntos de historia:</b> 3
<b>Descripción:</b> Historia de usuario para añadir la documentación acerca de clasificar series temporales a la memoria.			
<b>Tareas:</b> <ul style="list-style-type: none"><li>· Explicar investigación clasificación</li><li>· Añadir información a Home de WebApp</li></ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"><li>· </li></ul>			
<b>Observaciones:</b> Depende de HU023 debido a que es necesario el desarrollo antes de la documentación.			

Medidas de complejidad (DOC)			
<b>Identificador:</b> HU030	<b>Prioridad:</b> 1	<b>Iteración:</b> 1	<b>Puntos de historia:</b> 3
<b>Descripción:</b> Historia de usuario para añadir la documentación acerca de medidas de complejidad a la memoria.			
<b>Tareas:</b> <ul style="list-style-type: none"><li>· Descripción de medidas de complejidad.</li><li>· Añadir información a Home de WebApp.</li></ul>			
<b>Pruebas de aceptación:</b> <ul style="list-style-type: none"><li>· </li></ul>			
<b>Observaciones:</b>			

De nuevo se vuelven a dividir las historias de usuario por Sprint. En esta fase cada vez que se finalice un Sprint se procederá a liberar una nueva versión del producto ya que como se comenta en el apartado de software usado durante el desarrollo ahora haremos uso de un servidor de integración continua con ejecución de Tests y Build.

<b>Sprint:</b> 9	<b>Inicio:</b> 12/12/2016
Versión	Alpha-2.1.0
Identificador	Historia de usuario
HU035	Medidas de complejidad (DOC)
HU024	Clasificación de serie temporal (DOC)

<b>Sprint:</b> 10	<b>Inicio:</b> 26/12/2016
Versión	Alpha-2.2.0
Identificador	Historia de usuario
HU022	Clasificar series temporales (Mejoras)

<b>Sprint:</b> 11	<b>Inicio:</b> 09/01/2017
Versión	Alpha-2.3.0
Identificador	Historia de usuario
HU018	Misceláneos 1
HU020	Misceláneos 3
HU025	Subir serie temporal (Mejoras)

<b>Sprint:</b> 12	<b>Inicio:</b> 23/01/2017
Versión	Beta-1.0.0
Identificador	Historia de usuario
HU026	Listar series temporales con paginación
HU019	Misceláneos 2

<b>Sprint:</b> 13	<b>Inicio:</b> 06/02/2017
Versión	Beta-1.0.0 (No hay desarrollo)
Identificador	Historia de usuario
HU023	Documentación General (DOC)

## 2.3. Diagramas

En esta sección se mostrarán los diagramas de diseño utilizados en el sistema. Primero se va a mostrar un diagrama sobre el sistema completo.

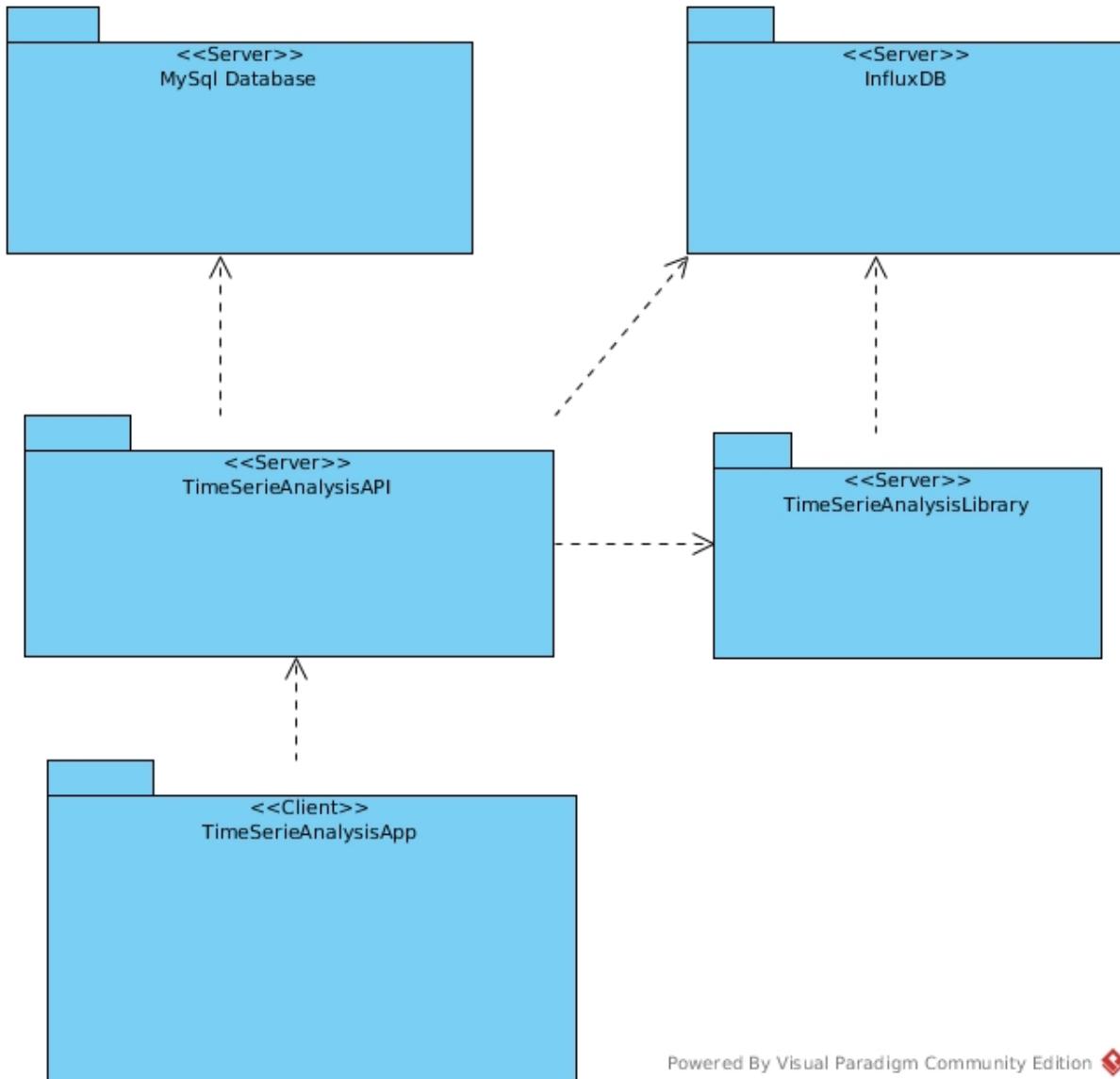


Imagen 1: Diagrama de módulos del sistema completo.

A continuación se mostrarán todos los diagramas del sistema comenzando desde la base de datos SQL, y continuando con todos los módulos del sistema, en el caso de TimeSeriesAnalysisLibrary también se mostraran los subdiagramas de clases de sus paquetes R:

- TimeSeriesAnalysisLibrary
- TimeSeriesAnalysisAPI
- TimeSeriesAnalysisApp

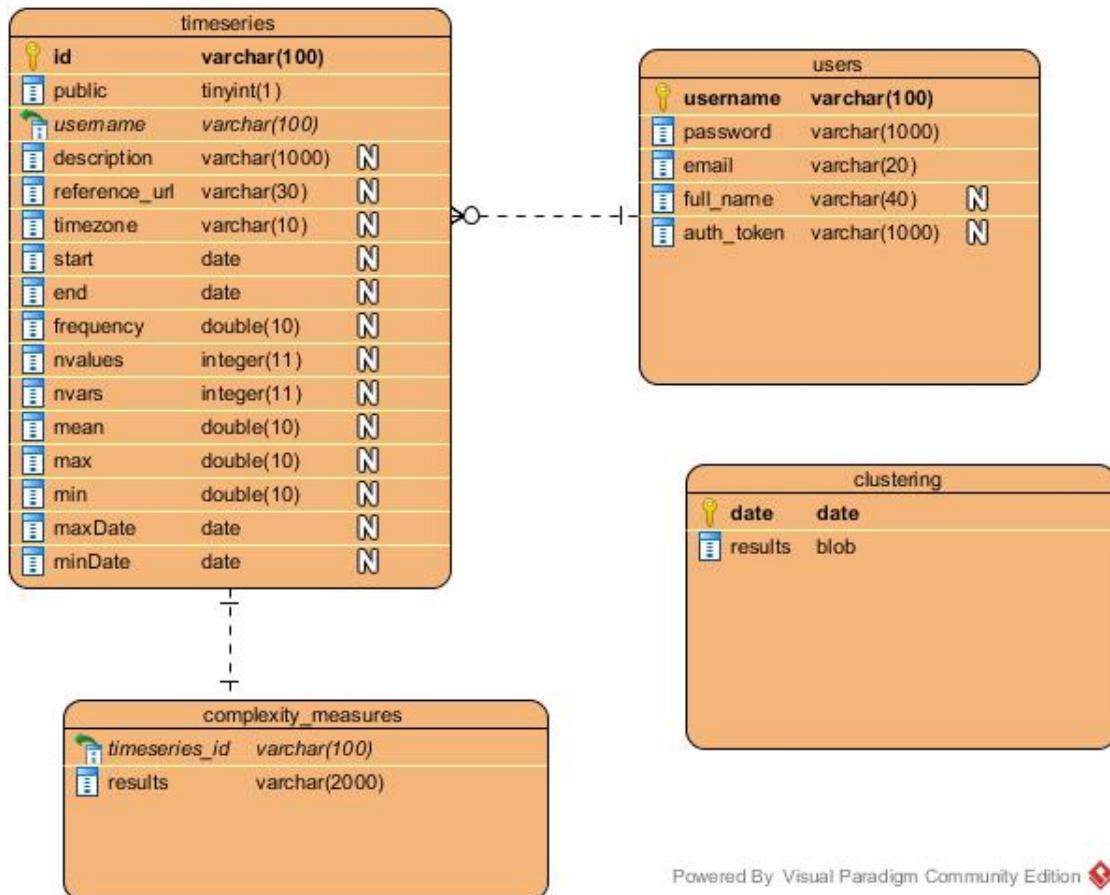


Imagen 2: Diagrama de datos SQL

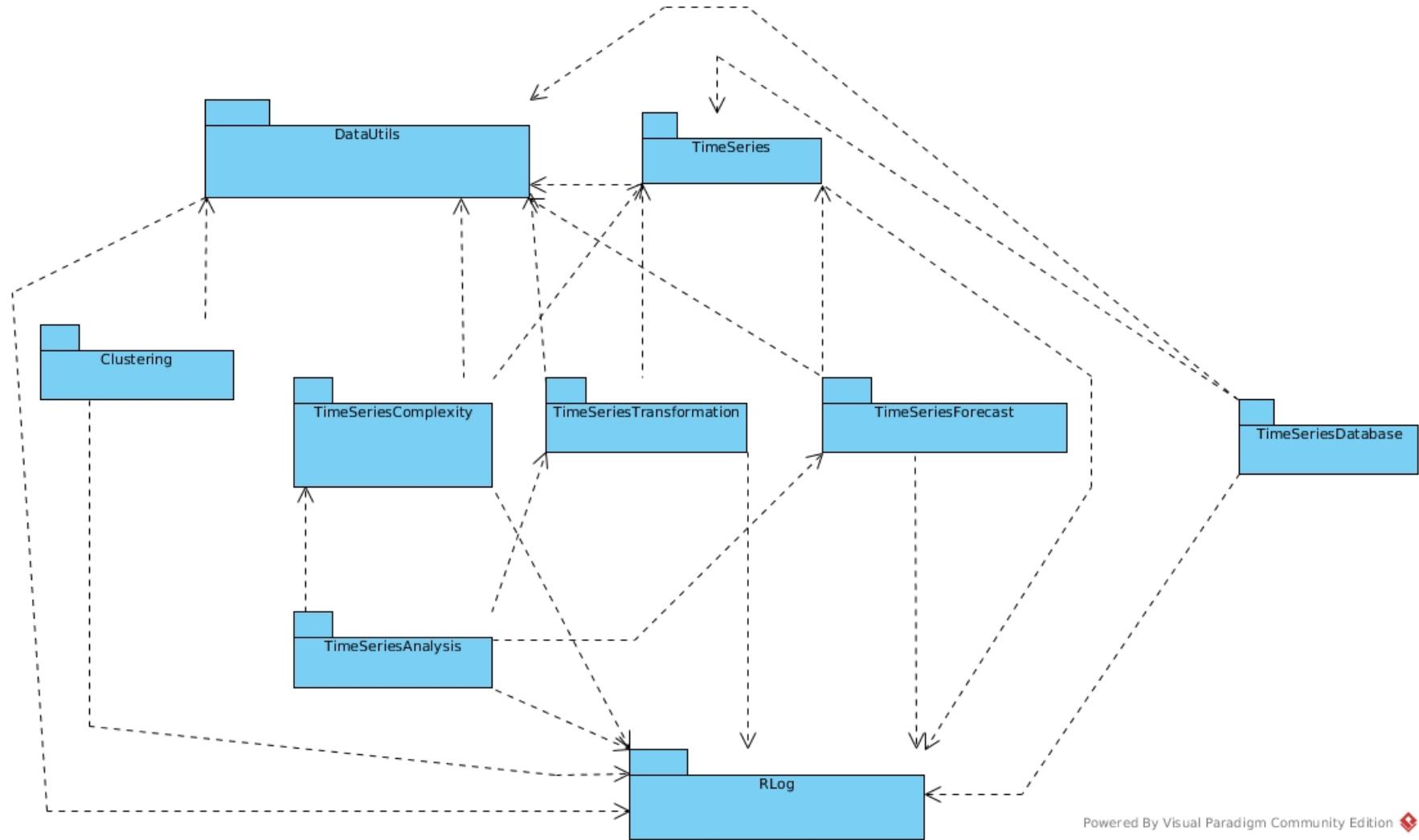


Imagen 3: Diagrama de paquetes R (TimeSeriesAnalysisLibrary)

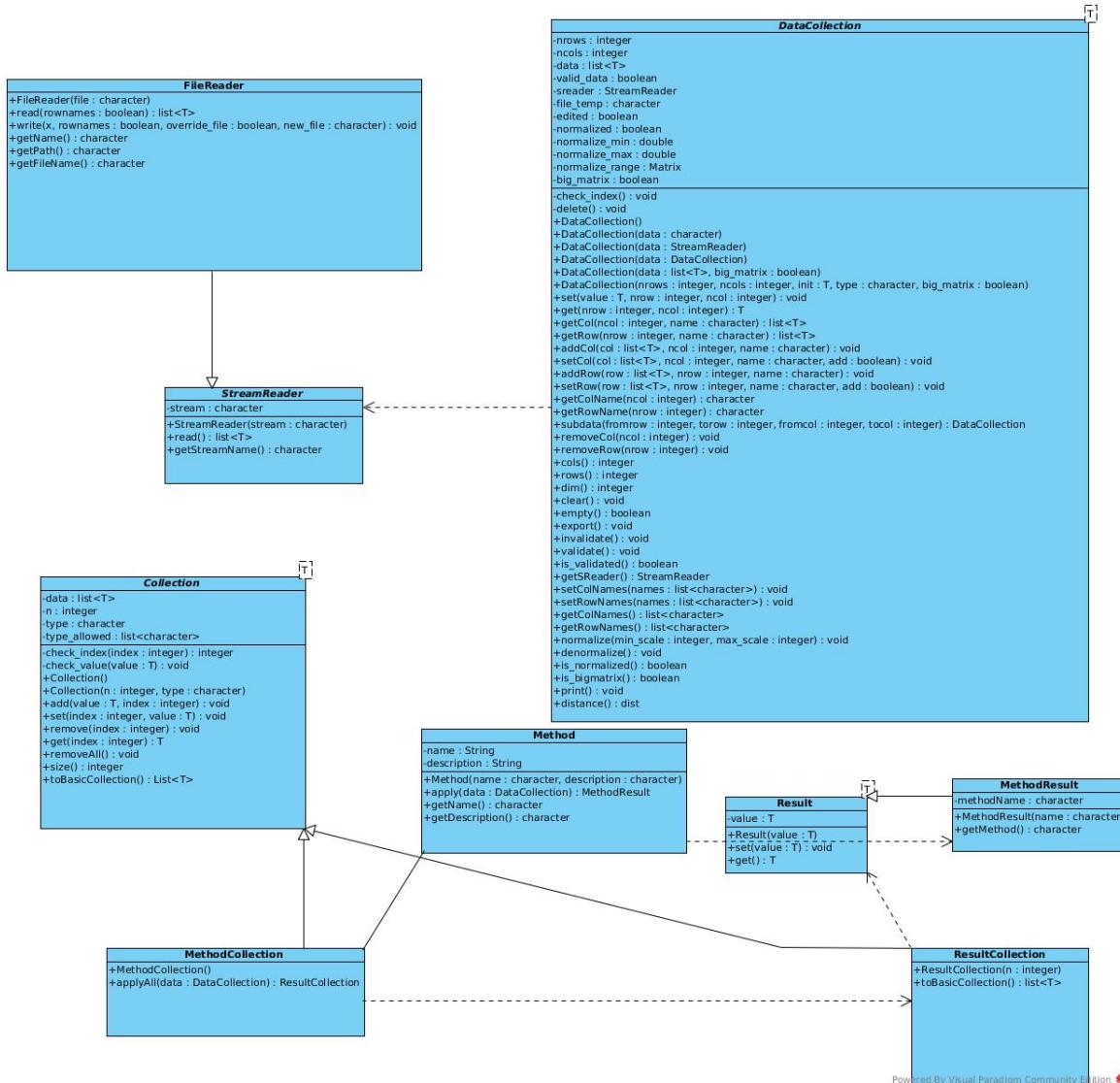


Imagen 4: Diagrama de clases del paquete R DataUtils.

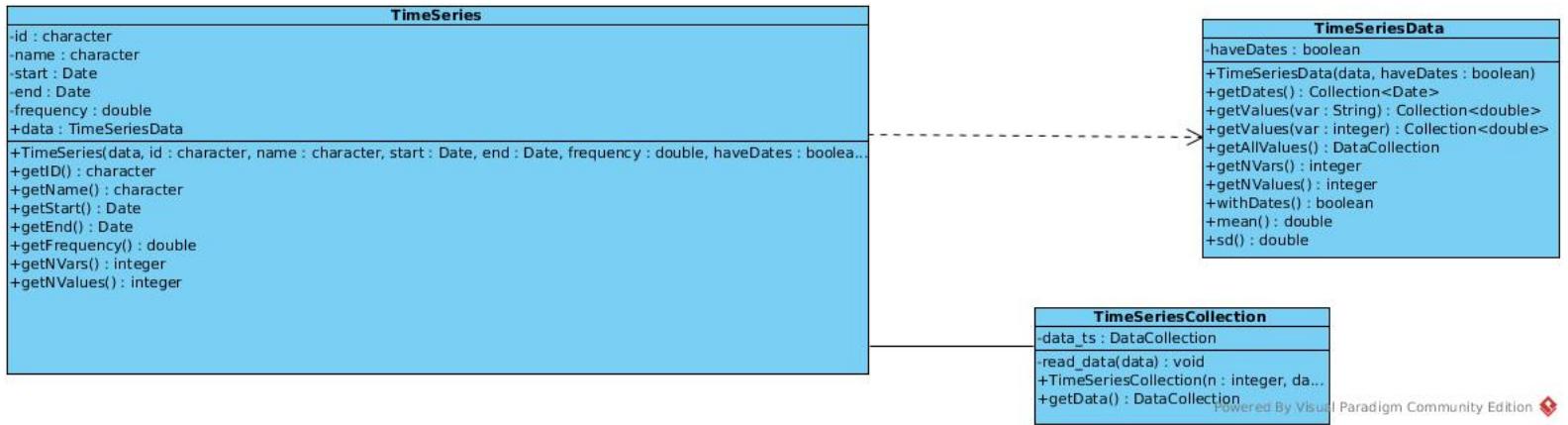


Imagen 5: Diagrama de clases del paquete R TimeSeries.

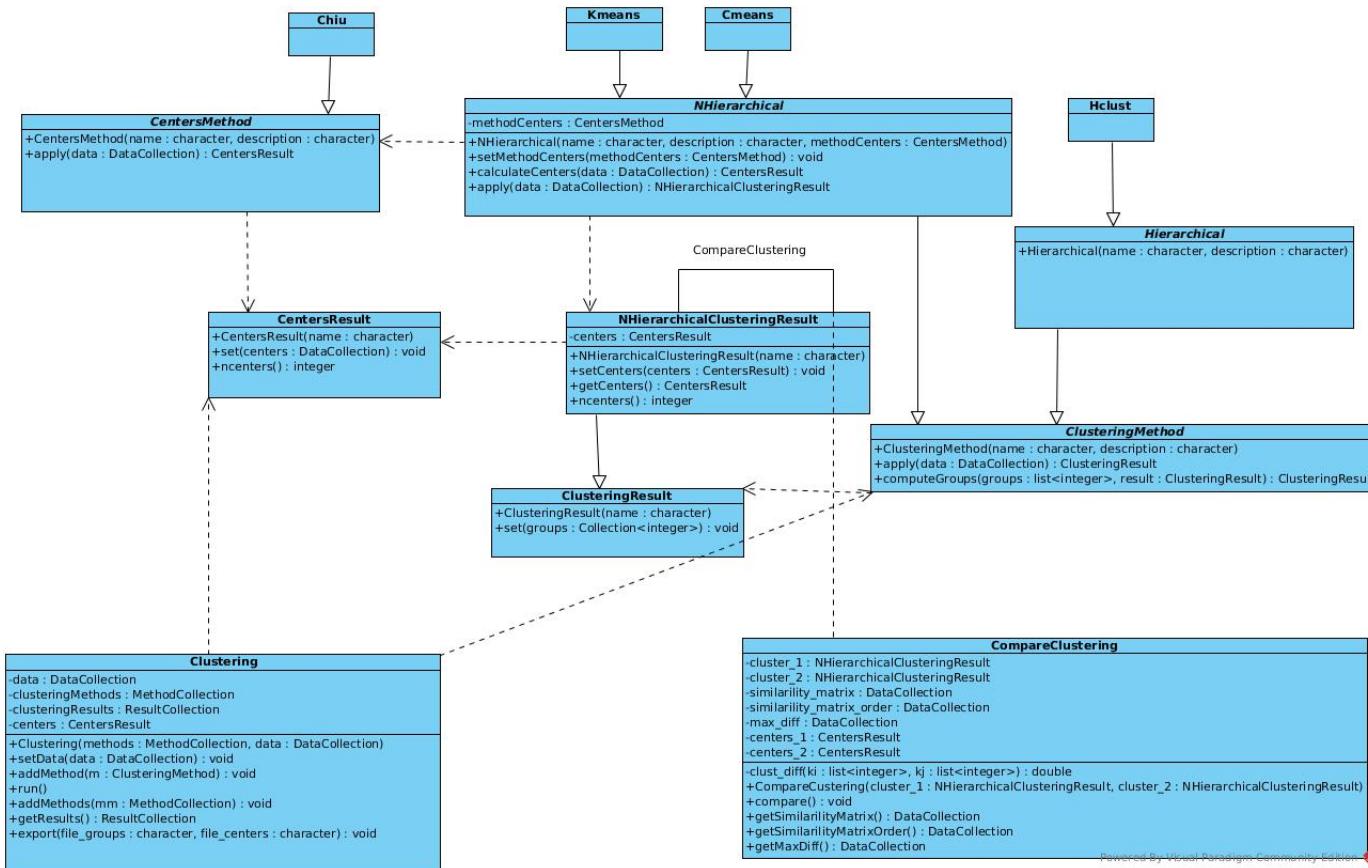


Imagen 6: Diagrama de clases del paquete R Clustering.

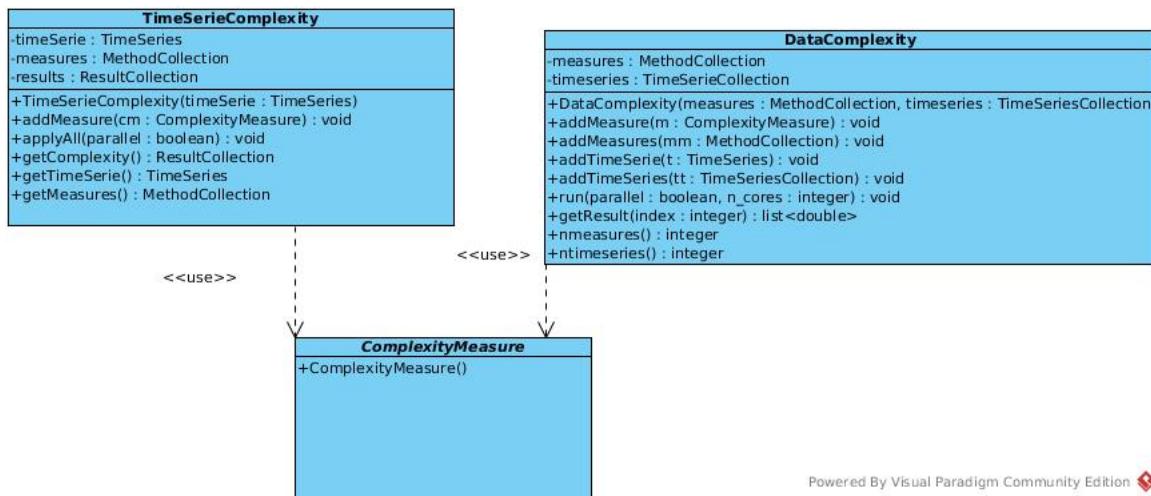


Imagen 7: Diagrama de clases del paquete R TimeSeriesComplexity.

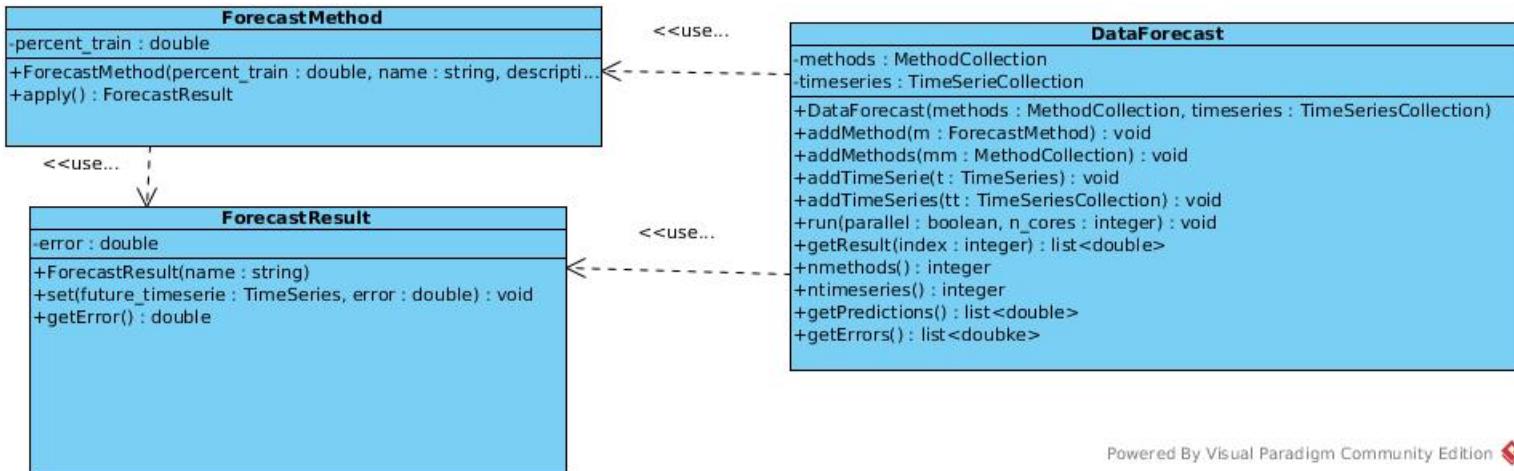


Imagen 8: Diagrama de clases del paquete R TimeSeriesForecast.

31

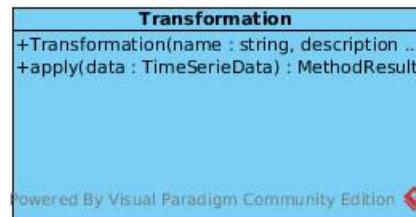


Imagen 9: Diagrama de clases del paquete R TimeSeriesTransformation.

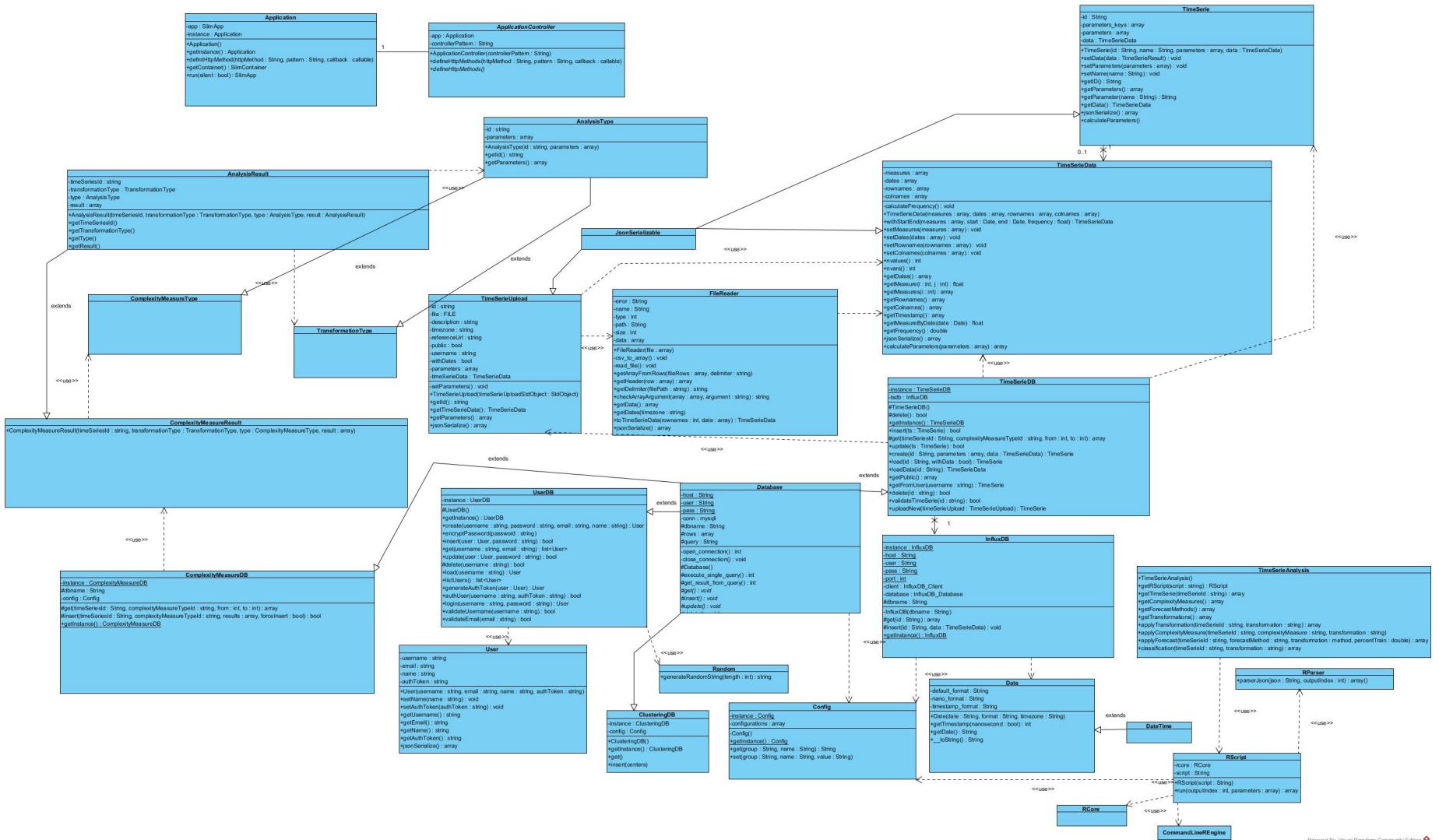
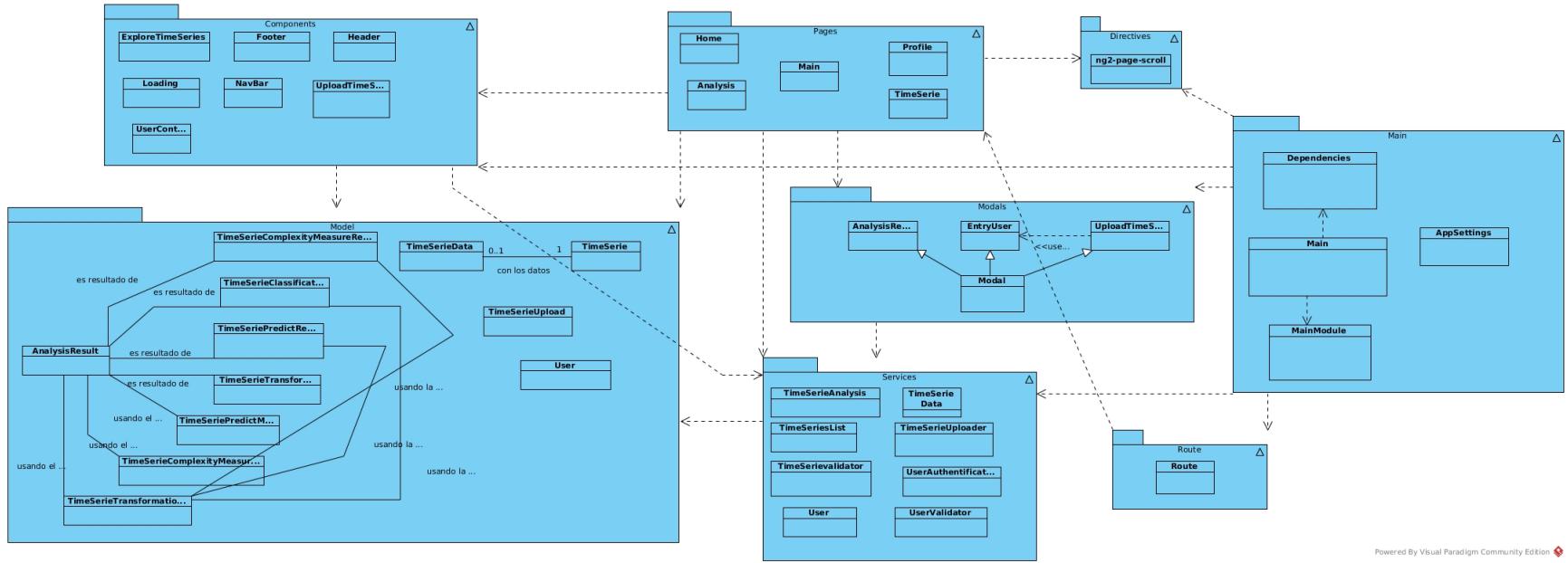


Imagen 10: Diagrama de clases API PHP (TimeSeriesAnalysisAPI)



Powered By Visual Paradigm Community Edition

CC

Imagen 11: Diagrama de clases Angular Aplicación Web. (TimeSeriesAnalysisApp)

## 2.4. Interfaces de usuario



## The basis of the project



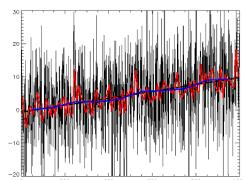
Time series. Its use is more important than it seems.

The time series data sets are used in many fields of research. Our intention is to provide a system to help us understand.



Time Serie Analysis. Intelligent system for analysis, classification and time series prediction.

With our system you can upload files containing time series data and in an automatic way you describe your time series, drawing properties, classify in a group and forecast with an error rate.



**Application features.** Simple and easy to use.  
With efficient implementation and an adapted and simple interface,  
any user without knowledge of time series can use it and get correct  
results.



Imagen 12: Página principal

Desde la página principal se inicia la aplicación con los dos botones superiores (Start Analysis App). Y se mueve por la página seleccionando los diferentes enlaces de la barra de navegación.

The screenshot shows the main interface of the application. At the top right are 'Log in' and 'Sign up' buttons. On the left, there is a circular arrow icon. The central area has two sections: 'Explore public time series' containing a table with 10 rows of data, and 'Upload new time serie' with a large blue cloud icon and an upward arrow.

ID	Values	Variables	Uploaded by
Fred_TOTALSL2	869	1	jmorenov
Test10	629	1	jmorenov
Test11	629	1	jmorenov
Test12	25789	1	jmorenov
Test13	25789	1	jmorenov
Test14	25789	1	jmorenov
Test15	25789	1	jmorenov
Test16	25789	1	jmorenov
Test17	25789	1	jmorenov
Test18	25789	1	jmorenov
Test20	25789	1	jmorenov

Imagen 13: Página de inicio de la aplicación.

En esta página se puede iniciar sesión o registrarse desde los dos botones superiores, estos botones abren un modal (Imágenes 14 y 15). A la izquierda está una flecha para volver a la página de inicio y la lista de series temporales públicas, seleccionando cualquiera de ellas se abrirá la página de serie temporal (Imagen 17). Y a la derecha el botón para subir una nueva serie temporal, al hacer click se abrirá otro modal con las diferentes opciones (Imagen 16).

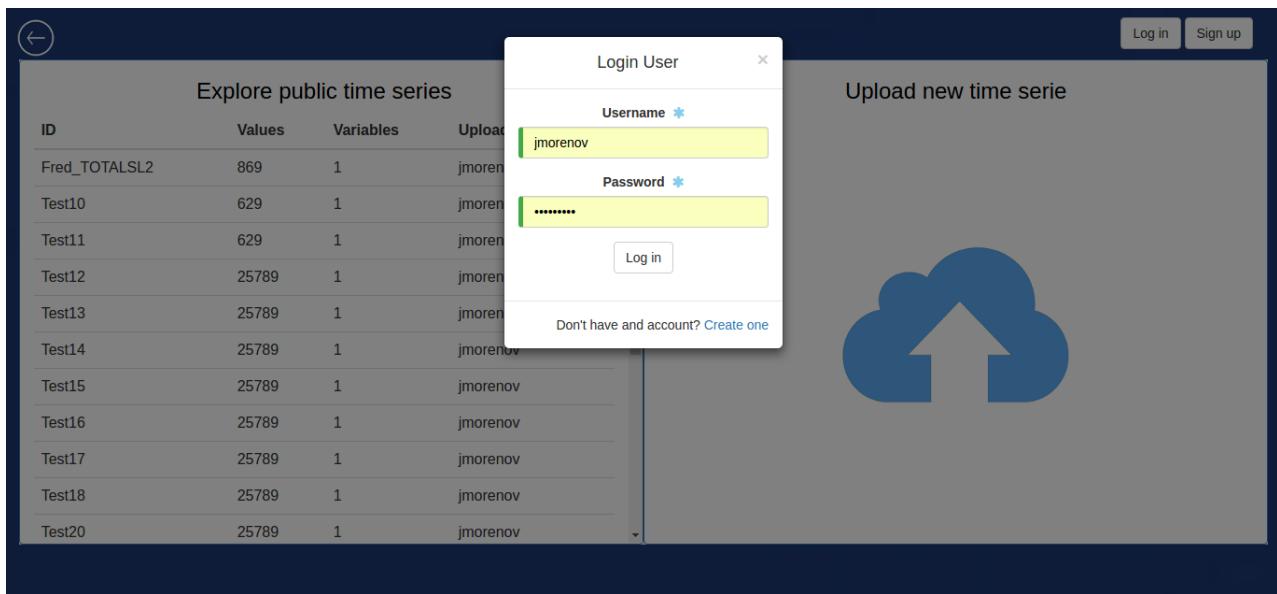


Imagen 14: Modal de inicio de sesión.

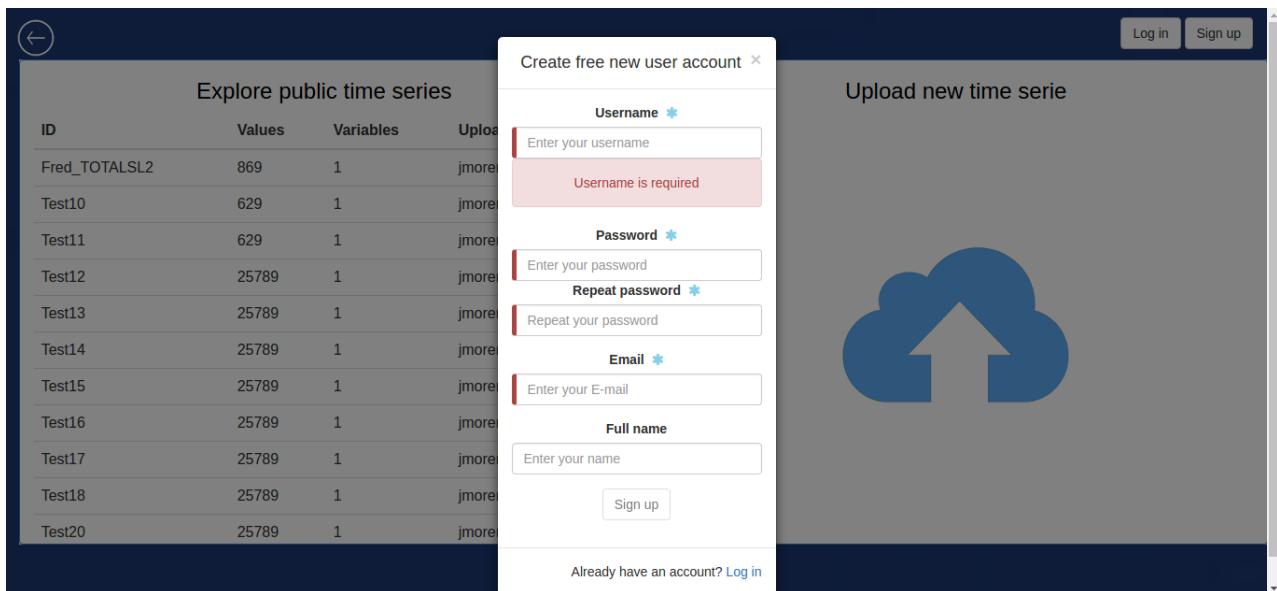


Imagen 15: Modal de registro de usuario.

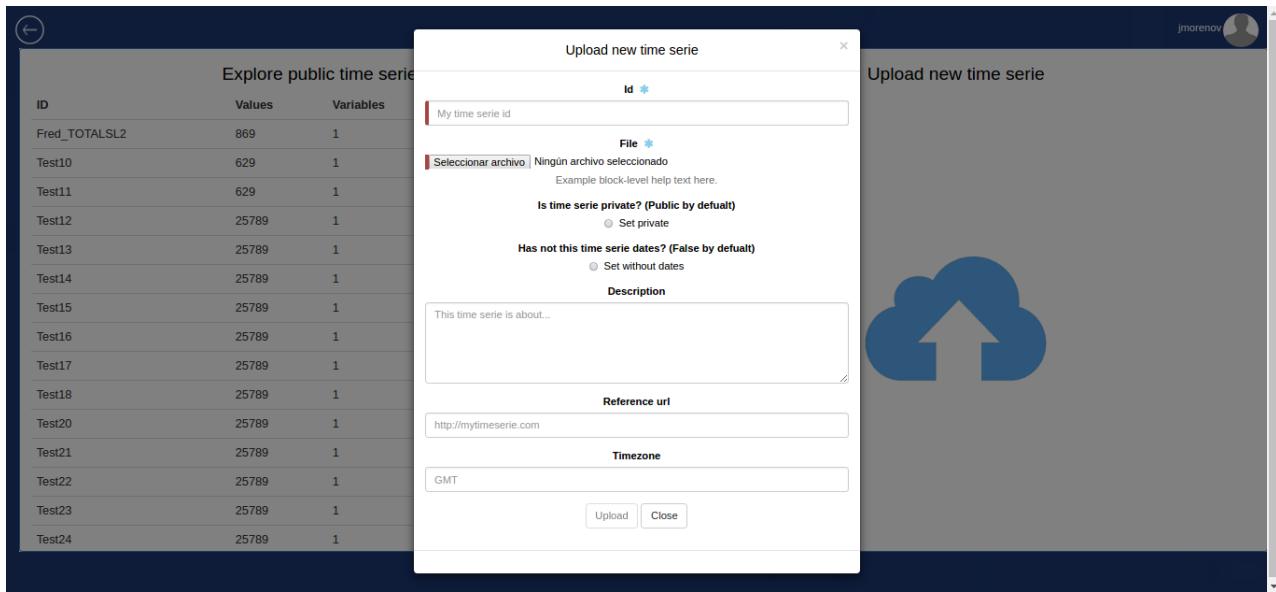


Imagen 16: Modal para subir nueva serie temporal.

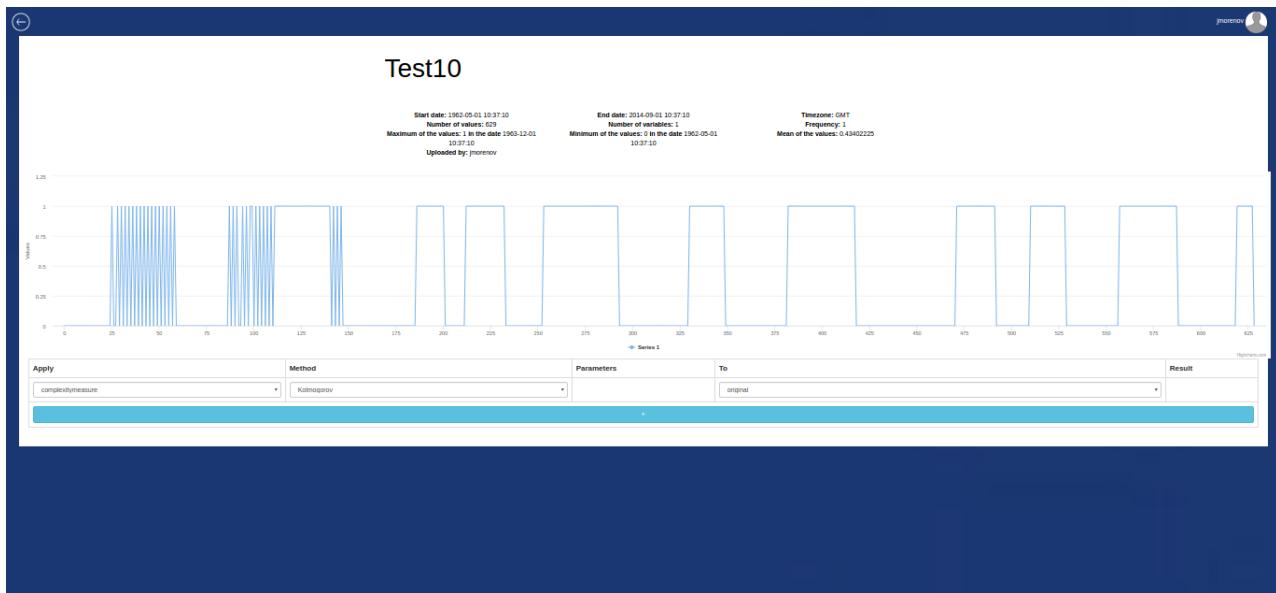


Imagen 17: Página de serie temporal.

En la página de serie temporal nos encontramos con todas las propiedades básicas de la misma, una gráfica con sus datos y la tabla para aplicar análisis al final. Si se hace click en la imagen de usuario en la esquina superior derecha se abrirá un dropdown (Imagen 18).

The screenshot shows a user interface for exploring public time series. On the left, there is a table titled "Explore public time series" with columns: ID, Values, Variables, and Uploaded by. The table contains 10 rows of data. On the right, there is a section titled "Upload new time serie" featuring a large blue cloud icon with an upward arrow. In the top right corner, there is a user profile icon labeled "jmorenov" and a dropdown menu with options "My Profile" and "Logout".

ID	Values	Variables	Uploaded by
Fred_TOTALSL2	869	1	jmorenov
Test10	629	1	jmorenov
Test11	629	1	jmorenov
Test12	25789	1	jmorenov
Test13	25789	1	jmorenov
Test14	25789	1	jmorenov
Test15	25789	1	jmorenov
Test16	25789	1	jmorenov
Test17	25789	1	jmorenov
Test18	25789	1	jmorenov
Test20	25789	1	jmorenov

Imagen 18: Dropdown con las opciones de usuario (Usuario logueado).

En este dropdown se puede navegar hacia la página de perfil de usuario (Imagen 19) o cerrar sesión.

The screenshot shows a user profile page for "jmorenov". It features a profile picture placeholder, the username "jmorenov", the email "hola@gmail.com", and the full name "Javier Moreno". There is a "Edit profile" button with a wrench icon. Below this, there is a section titled "My time series" with a table showing a list of time series entries. The table has columns: ID, Values, Variables, and Status. The entries include "FRED\_4BIGEUROREC", "Fred\_TOTALSL2", "Tes5", "Test", and "Test10", each with their respective values, variables, and status (Private or Public).

ID	Values	Variables	Status
FRED_4BIGEUROREC	1166	1	Private
FRED_4BIGEUROREC2	629	1	Private
Fred_TOTALSL2	869	1	Public
Tes5	8	1	Private
Test	8	1	Private
Test10	629	1	Public

Imagen 19: Página de perfil de usuario.

En esta página se muestra el perfil de usuario junto con una lista de sus series temporales, públicas y privadas, si son privadas solo el propio usuario puede acceder a ellas, obteniendo un mensaje de error si accediera un usuario sin

permisos. Además tenemos la opción de editar el perfil si se hace click en el elemento de la derecha (Imagen 20).

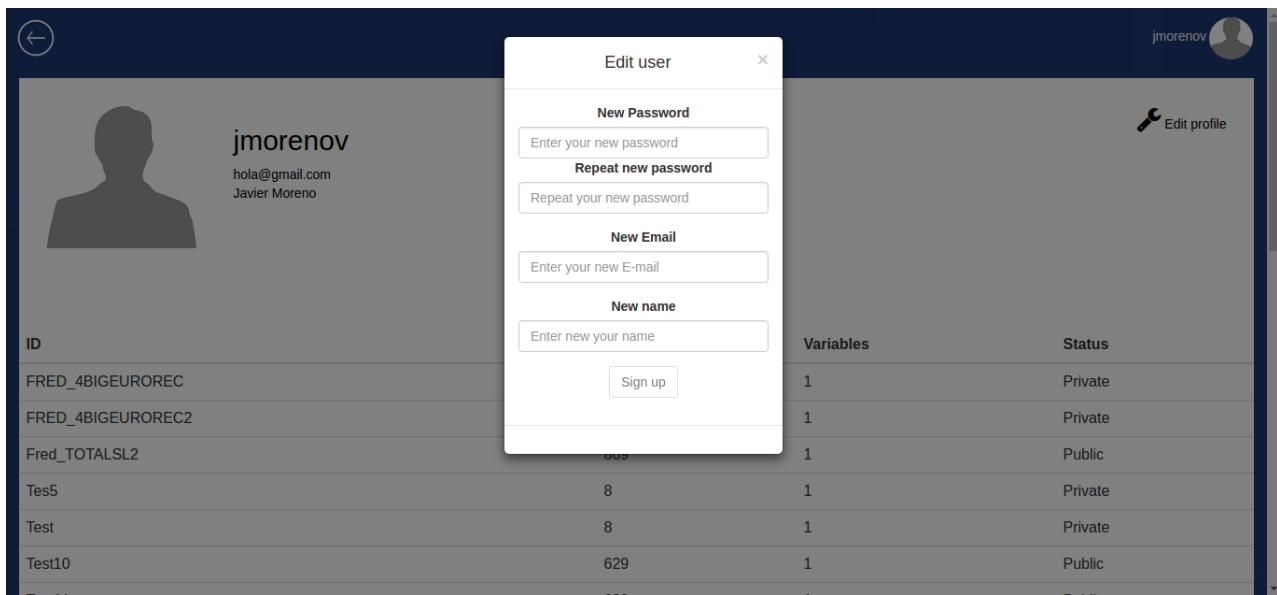


Imagen 20: Modal de editar perfil de usuario.

## 2.5. Herramientas utilizadas

En este desarrollo se han utilizado diversas herramientas de apoyo como bibliotecas externas que nos ofrecen distintas funcionalidades y software de apoyo al desarrollo. Este software usado ha sido:

- **Git [58]**: Software de control de versiones donde se ha ido subiendo el código desarrollado.
- **Docker [56]**: Software que ofrece contenedores para el despliegue de aplicaciones. Es de gran utilidad para el despliegue en la nube. En este proyecto se han utilizado un total de 10 contenedores Docker y gracias a ellos el sistema completo puede hacerse distribuido muy fácilmente.
- **RStudio [63]**: IDE para el desarrollo en R, que dispone además de un gestor de paquetes.
- **Phpstorm [17]**: IDE para el desarrollo en PHP.
- **PyCharm [18]**: IDE para el desarrollo en Python.
- **Youtrack [20]**: Aplicación web, montada sobre Docker, para ayudar al desarrollo en Scrum.
- **Teamcity [19]**: Aplicación web, montada también sobre Docker, que lleva el control del código y la integración continua. Ha sido muy útil para controlar fallos tempranos de compilación de código o ejecución de tests.
- **Npm [47]**: Gestor de paquetes para Angular.
- **Composer [5]**: Gestor de paquetes para PHP.
- **Visual paradigm [31]**: Software para el diseño de diagramas.
- **InfluxDB [14]**: Este software es una base de datos de series temporales a la cual se accede mediante consultas http. Almacena las series temporales como datos y fechas, pudiendo añadir además Tag y teniendo un identificador único. De no definir fechas al insertar la serie temporal las fechas que se

añaden son el timestamp del momento de inserción. Además ofrece bibliotecas muy útiles, en este caso se ha usado una para acceder mediante PHP (influxdb-php [15]) con la que se accede desde la API.

- **Apache** [64]: Software para el desarrollo de servidores web.
- **PhpMyAdmin** [62]: Aplicación web para la gestión de datos en un servidor sql.

El desarrollo de la biblioteca de análisis, TimeSeriesAnalysisLibrary, ha sido realizado en RStudio y se ha hecho uso de diversos paquetes, siendo los más importantes:

- R6 [4]: Ofrece el desarrollo orientado a objetos en R.
- entropy [10]: Medidas de complejidad de entropía.
- parallel [33]: Ejecución en paralelo.
- testthat [43]: Métodos para testear código en R.
- forecast [13]: Predicción de series temporales.
- xts [16]: Métodos para interactuar con series temporales.
- Rcpp [7]: Desarrollo de funciones para R en C++.
- RcppParallel [34]: Funciones C++ en paralelo.
- bigmemory [24]: Genera archivos temporales para trabajar con archivos muy grandes.

La API, TimeSeriesAnalysisAPI, se ha desarrollado usando PhpStorm y Composer, y los paquetes utilizados han sido:

- guzzle [9]: Ofrece funcionalidad para realizar peticiones Http.
- influxdb-php [15]: Desarrollado para hacer peticiones a una base de datos Influxdb.
- slim [36]: Framework para desarrollar una API en Php.

- `phpunit` [32]: Framework que ofrece la funcionalidad para ejecutar tests sobre Php.

Por último la aplicación web, TimeSeriesAnalysisWebApp, se ha desarrollado con PhpStorm (también ofrece desarrollo web javascript) y usando Npm como gestor de paquetes. Los paquetes más importantes han sido:

- `angular` [8]: Framework para el desarrollo web.
- `bootstrap` [3]: Framework que ayuda al desarrollo web ofreciendo gran diversidad de estilos css.
- `highcharts` [12]: Ofrece funcionalidad para generar gráficos.
- `karma` [25]: Ofrece funcionalidad para ejecutar tests.
- `ngx-modal` [29]: Ofrece funcionalidad para mostrar modales.

### **3. Biblioteca de análisis (TimeSeriesAnalysisLibrary)**

La biblioteca de análisis (TimeSeriesAnalysisLibrary) es un conjunto de métodos que ofrecen funcionalidad que se puede aplicar a series temporales, la principal es la de analizar series temporales. La idea es que todo sea lo más transparente al usuario posible, es decir, el usuario seleccionaría un archivo y la biblioteca se encargaría del resto.

La biblioteca ofrece la siguiente funcionalidad para aplicar sobre series temporales:

- Medidas de complejidad
- Transformaciones
- Clustering
- Predicción
- Clasificación

Todos estos métodos de análisis pueden ser aplicados individualmente o en grupos, por ejemplo todas las medidas de complejidad a la vez sobre una o sobre un conjunto de series temporales. Este tipo de cálculos son útiles para el uso de Clustering que será explicado más adelante.

Vamos a analizar uno a uno los métodos generales, o módulos de la biblioteca, empezando por las medidas de complejidad.

#### **3.1. Medidas de complejidad**

Una medida de complejidad es un cálculo que se aplica sobre un conjunto de datos, en este caso series temporales, y devuelve como resultado la complejidad de los datos de ésta. Estos resultados son una de las partes más importantes de este proyecto ya que se usan para clasificar series temporales respecto a su complejidad, en el Clustering.

Las medidas de complejidad, que implementa la biblioteca son las siguientes:

### - Kolmogorov[35][60]

La complejidad Kolmogorov  $K(x)$  de un objeto  $x$  es la longitud, en bits, de el programa más pequeño (en bits) que puede ejecutarse en una Máquina de Turing Universal [61] ( $U$ ), devuelve  $K(x)$  y finaliza la ejecución. Esta medida fue desarrollada independientemente por Andrey N. Kolmogorov en el final de los años 60.

La complejidad Kolmogorov de una serie temporal  $x_i$ ,  $i=1,2,3,4,\dots,N$ ; puede ser calculada de la siguiente forma [6]:

Paso 1: Codificar la serie temporal construyendo una secuencia  $s$ , formada con los caracteres 0 y 1, escrital:  $s(i)$ ,  $i=1,2,3,4,\dots,N$ ; según la regla:

$$s(i) = \begin{cases} 0 & x_i < x_* \\ 1 & x_i \geq x_* \end{cases}$$

Donde  $x_*$  es un umbral, el valor medio de la serie temporal se usa con frecuencia como el umbral.

Paso 2: Calcular el contador de complejidad  $C(N)$ , el cual está definido como el número mínimo de patrones distintos contenidos en una secuencia de caracteres.

### - Lempel-Ziv[26]

Esta medida de complejidad está basada en la idea de Kolmogorov, Lempel y Ziv desarrollaron el algoritmo (LZA), el cuál es con frecuencia usado en la evaluación de la aleatoriedad de las secuencias finitas como una medida de su desorden.

Primero se calcula la complejidad de Kolmogorov  $K(x)$  de un objeto  $x$  y luego la complejidad normalizada  $C_k(N)$ , la cuál está definida por la ecuación:

$$C_k(N) = \frac{c(N)}{b(N)} = c(N) \frac{\log_2 N}{N} \quad (1)$$

### - Aproximation Entropy

Aproximation Entropy (ApEn) es una medida de complejidad utilizada para cuantificar la cantidad de regularidad y la imprevisibilidad de las fluctuaciones sobre los datos de series temporales. La presencia de patrones repetitivos de fluctuación en una serie temporal la hace más predecible que una serie temporal en la que tales patrones están ausentes.

ApEn refleja la probabilidad de que patrones de observaciones similares no sean seguidos por otras observaciones similares también. Una serie temporal que contiene muchos patrones repetitivos tiene un ApEn relativamente pequeño, mientras que una menos predecible tiene uno mayor.

### - Sample Entropy

Sample entropy (SampEn) es una modificación de approximate entropy, utilizada ampliamente para evaluar la complejidad de una señal temporal fisiológica, diagnosticando así enfermedades. Al igual que ApEn, SampEn es una medida de complejidad pero no incluye patrones auto-similares como hace ApEn.

### - Permutation Entropy

Permutation entropy (PE) es una medida de complejidad, para series temporales arbitrarias, basada en el análisis de patrones de permutación.

Entrada: Dada una serie temporal  $X = \{x_i\}_{i=1\dots N}$

Hay que determinar la longitud de la ventana deslizante  $n$  y deslizar la serie temporal de acuerdo a ella, calcular PE  $h_n$  de orden  $n$ , repetir para varios  $n$  y al final calcular  $h_n$ , donde:

$$H(n) = - \sum p() \log p(\pi)$$

$$h_n = \frac{H(n)}{(n - 1)}$$

### - Shannon Entropy[42][65]

Definimos el tamaño de información  $H_0(A)$  de un conjunto A como el número de bits necesarios para codificar cada elemento de A independientemente.

$$H_0(A) = \log_2 |A|$$

Basada en esta idea nace Shannon Entropy (ShEn), llamada así por Claude Elwood Shannon [55], aunque su origen recae en Pauile y von Neumann.

La entropía de A es:

$$H(A) = - \sum_{i=1}^n p_i \log_2 p_i$$

Donde  $p_i$  es la probabilidad de que el carácter i aparezca en la secuencia.

- **ChaoShen Entropy**[38][11]

Medida de complejidad que calcula ShEn H de la variable aleatoria Y de las muestras observadas y, usando el método de Chao and Shen (2003).

El estimador de entropía Chao-Shen es un estimador Horvitz-Thompson (1952) aplicado al problema de la estimación de entropía, con una corrección de cobertura adicional según lo propuesto por Good (1953).

- **Dirichlet Entropy**[39]

El estimador de entropía de Dirichlet es un estimador bayesiano de plugin: en la definición de la entropía de Shannon las probabilidades de bin son reemplazadas por las respectivas estimaciones bayesianas de las frecuencias, usando un modelo con una probabilidad Dirichlet anterior y multinomial.

- **MillerMadow Entropy**[30][41]

Esta medida de complejidad estima la entropía de Shannon H de la variable aleatoria Y de las muestras observadas y, aplicando la corrección Miller-Madow a la entropía empírica.

En 1955 George Miller propuso una simple corrección al estimador naive plugin  $\hat{H}_N$  añadiendo el desplazamiento constante en la expresión de sesgo:

$$\hat{H}_M = \hat{H}_N + \frac{K - 1}{2n}$$

Esto es una mejora sobre el estimador plugin, pero el desplazamiento añadido no depende de la distribución, sino sólo del tamaño de la muestra. Una variante mejora esta corrección, actua sobre el caso del alfabeto infinito y es llamada: estimador Miller-Madow. En él la cantidad K es estimada desde los datos.

- **Shrink Entropy**[40]

Shrink Entropy es una medida de complejidad basada en el estimador shrinkage que es un estimador James-Stein. Es en esencia un estimador dirichlet, donde pseudocount es estimada desde los datos.

## 3.2. Transformaciones

En la biblioteca están incluidas varias transformaciones sobre datos. A continuación se muestran estas transformaciones con sus fórmulas respectivas. Sea  $X$  una serie temporal:

- **Transformación Cúbica**

$$TransCub(X) = X^3$$

- **Transformación Logarítmica Natural**

$$TransLN(X) = \log(X)$$

- **Transformación Logarítmica Base 10**

$$TransL10(X) = \log_{10}(X)$$

- **Transformación Logarítmica Base 2**

$$TransL2(X) = \log_2(X)$$

- **Transformación Exponencial**

$$TransExp(X) = \exp(X)$$

- **Transformación Raiz Cuadrada**

$$TransSqrt(X) = \sqrt{X}$$

- **Transformación Valor Absoluto**

$$TransAbs(X) = |X|$$

- **Transformación Seno**

$$TransSin(X) = \sin(X)$$

- **Transformación Arco Seno**

$$TransASin(X) = \arcsin(X)$$

- **Transformación Coseno**

$$TransCos(X) = \cos(X)$$

## - Transformación Arco Coseno

$$TransACos(X) = \text{acos}(X)$$

Son transformaciones básicas pero muy útiles para ver como se comportan determinadas series temporales.

## 3.3. Clustering

Una definición básica de Clustering de datos es: Un algoritmo de agrupamiento (en inglés, clustering) es un procedimiento de agrupación de una serie de vectores de acuerdo con un criterio. Esos criterios son por lo general distancia o similitud. [54]. En general es encontrar subconjuntos de datos, dentro de un conjunto, que sean similares entre sí. En este caso para poder aplicar determinados métodos de análisis de series temporales a un subconjunto y otros a otro, consiguiendo unos mejores resultados.

TimeSeriesAnalysisLibrary también implementa métodos de clustering, y funcionalidad útil sobre clustering como el cálculo de centros con el método de Chiu. Esta basado en el cálculo de potenciales, si tenemos un conjunto de datos  $M \times m$ ,  $x_i (i = 1, \dots, m)$ , se calcula el potencial para cada dato y secuencialmente se selecciona el dato restante de mayor potencial. Se define el potencial del dato  $x_i$ :

$$P_i = \sum_{j=1}^M \exp\left(-\frac{4\|x_i - x_j\|^2}{r_a^2}\right)$$

Siendo  $\|x_i - x_j\|$  la distancia Euclídea entre  $x_i$  y  $x_j$ , y  $r_a$  es un parámetro positivo que controla el decrecimiento del radio del potencial. Este algoritmo está en: [1, Chapter 6.1.1]

El tipo de clustering se divide en dos: jerárquico y no-jerárquico o de particionamiento. El clustering Jerárquico es un método de análisis de grupos puntuales, el cual busca construir una jerarquía de grupos, básicamente genera un árbol de decisión sobre los datos [53]. El clustering no-jerárquico, a partir de unos centros dados (método de Chiu), genera particiones del conjunto de datos.

La biblioteca implementa métodos de los dos tipos, en primer lugar los jerárquicos son:

- **MClust [37]**: Está dentro del tipo de los jerárquicos pero pertenece a un subconjunto denominado clustering basado en modelos en donde los datos se considera que vienen de una distribución que es la mezcla de dos o más componentes (clusters). Cada componente  $k$  (cluster) está modelado por la normal o la distribución Gaussiana la cual está caracterizada por los parámetros:  $\mu_k$  vector medio,  $\Sigma_k$  matriz de covarianza, y una probabilidad asociada al componente, cada punto tiene una probabilidad de pertenecer a cada cluster.
- **HClust [27]**: Este tipo de clustering es por definición jerárquico, pero en esta librería podría ser considerado de particionamiento debido a que aunque se usa su algoritmo principal para generar el árbol de decisión este árbol es podado usando el número de centros de clustering  $k$ , que es la base de los clustering de particionamiento.

El algoritmo funciona de la siguiente forma: Selecciona cada punto y lo pone en un cluster propio, selecciona los dos clusters más cercanos y los combina en uno; repite este proceso hasta que todos los puntos (los más cercanos) estén en un mismo cluster.

Y los no-jerárquicos:

- **Fuzzy c-means [57]**: Los algoritmos Fuzzy c-Means son algunos de los principales algoritmos utilizados en el agrupamiento difuso y pertenecen a una clase de algoritmos basados en funciones objetivo. Estos algoritmos definen un criterio de agrupamiento en la forma de una función objetivo que depende de la partición difusa. El procedimiento, en sentido general, consiste en minimizar iterativamente esta función hasta obtener una partición difusa óptima.
- **K-means [59]**: Probablemente el algoritmo de agrupamiento más conocido. Tiene como objetivo la partición de un conjunto de  $n$  observaciones en  $k$  grupos en el que cada observación pertenece al grupo cuyo valor medio es más cercano.

### 3.4. Predicción

La predicción de series temporales es de las partes de más importancia de este trabajo. La biblioteca cuenta con varios métodos de predicción:

- **ARIMA [45]:** Probablemente el método de predicción de series temporales más conocido. Modelo autorregresivo integrado de promedio móvil (del inglés ARIMA), es un modelo estadístico que utiliza variaciones y regresiones de datos estadísticos con el fin de encontrar patrones para una predicción hacia el futuro. Se trata de un modelo dinámico de series temporales, es decir, las estimaciones futuras vienen explicadas por los datos del pasado y no por variables independientes. Fue desarrollado a finales de los sesenta del siglo XX. Box y Jenkins (1976) lo sistematizaron.
- **Ets:** Es un modelo de espacio de suavizamiento exponencial. Basado en la clasificación de métodos como describe Hyndman (2008).
- **Holtwinters [51]:** El modelo Holt-Winters incorpora un conjunto de procedimientos que conforman el núcleo de la familia de series temporales de suavizamiento exponencial. A diferencia de muchas otras técnicas, el modelo Holt-Winters puede adaptarse fácilmente a cambios y tendencias, así como a patrones estacionales. En comparación con otras técnicas, como ARIMA, el tiempo necesario para calcular el pronóstico es considerablemente más rápido.
- **StructTS [2]:** Este método de predicción está basado en los modelos estructurales de series temporales. La idea básica de los modelos de series temporales estructurales es que se establecen como modelos de regresión en los que las variables explicativas son funciones del tiempo con coeficientes que cambian con el tiempo. Así, dentro de un marco de regresión, una tendencia simple sería modelada en términos de una constante y un tiempo con una perturbación aleatoria agregada. Este modelo es fácil de estimar usando mínimos cuadrados ordinarios, pero sufre la desventaja de que la tendencia es determinista.
- **Bats [22]:** Método de predicción basado en el modelo BATS, esto es un acrónimo de: Box-Cox transform, Arma errors, Trend and seasonal. Es un modelo que agrupa varios.

- **Stl [21]:** Este método de predicción consiste en aplicar un método de predicción no estacional a los datos desestacionalizados y re-estacionalizarlos utilizando el último año del componente estacional.

### 3.5. Clasificación

Esta es la parte más importante de todo el trabajo, y depende de los métodos explicados en los apartados anteriores para realizar los cálculos necesarios. El resultado que se quería obtener era una clasificación de series temporales, es decir, definir un conjunto finito de etiquetas que asignar a cualquier serie temporal que se requiriera. Con esto se conseguía generar conjuntos de series temporales con propiedades similares y que por lo tanto tendrían resultados parecidos cuando se les aplicaran métodos de predicción, o de cualquier tipo.

Para conseguir esta clasificación se usaron conjuntos grandes de series temporales (200000), a las que se les aplicaría todas las medidas de complejidad explicadas anteriormente.

Con esto tenemos una matriz de resultados y entonces es el momento de aplicar los métodos de clustering, que nos particionaran las series temporales, generando una clasificación previa. A partir de esta clasificación, una por cada método de clustering, faltaba por decidir que método de predicción genera mejores resultados para cada conjunto de series temporales. Este resultado final lo obtenemos aplicando todos los métodos de predicción a cada conjunto de series temporales y calculando el porcentaje medio de éxito método/partición de datos. Y con esto tendremos el método de mejor resultado para cada tipo de serie temporal.

Para clasificar una serie temporal se generan sus medidas de complejidad y se calcula la distancia a los centros, seleccionando la menor distancia; teniendo el centro al que pertenece obtenemos el mejor método de predicción.

## 4. Implementación

En esta sección se explicará la implementación de código que se ha realizado, mostrándola más detalladamente en las secciones de a continuación.

### 4.1. Despliegue en Cloud (DockersProject)

Con el fin de conseguir una aplicación distribuida en Cloud se ha usado el software Docker [56]. Este software nos permite construir contenedores que funcionan como servidores dentro de un servidor padre, se pueden configurar de cualquier forma y hay bastantes contenedores ya construidos por terceros y modificables.

Además para poder iniciar, reiniciar, parar o configurar cualquier contenedor de los que se han construido se hicieron scripts y para hacer estos scripts más reutilizables se ha desarrollado un pequeño proyecto llamado DockersProject.

DockersProject está escrito en python y se gestiona mediante comandos y sus argumentos son los argumentos del comando de ejecución, usa como archivo de configuración de los contenedores un archivo json con todo lo necesario: nombre, imagen, localización de la imagen, variables de entrada, puertos, etc. Este archivo se puede modificar fácilmente para añadir más contenedores o modificar los existentes y el comando es muy sencillo, por ejemplo: python DockersControl.py –start-all, este comando lanzaría todos los contenedores.

Los contenedores que se han usado han sido:

- TimeSeriesAnalysisWebApp: Este contenedor tiene instalado NPM [47] con todos los paquetes javascript necesarios para lanzar la aplicación web, así como scripts para actualizar el código y tener siempre el contenedor con la última versión de la aplicación. Además incluye una versión de Apache.
- TimeSeriesAnalysisAPI: Este contenedor tiene instalado composer [5] con los paquetes Php necesarios, así como los scripts para actualizar el código como en el contenedor anterior. También Apache [64], Php y TimeSeriesAnalysisLibrary con todos los paquetes R y los scripts de análisis.

- Mysql: Contenedor con un servidor mysql, incluye los scripts sql necesarios para el sistema.
- Phpmyadmin: Contenedor con una versión de Phpmyadmin [62] para acceder a los datos del servidor Mysql fácilmente.
- InfluxDB: Contenedor que contiene una versión de InfluxDB [14] con todas las series temporales almacenadas en el sistema.
- Teamcity: Contenedor con una versión de Teamcity [19].
- Youtrack: Contenedor con una versión de Youtrack [20].
- TeamcityAgent 1, 2 y 3: Contenedores con un agente de Teamcity cada uno, son usados para ejecutar el código sobre la aplicación Teamcity.

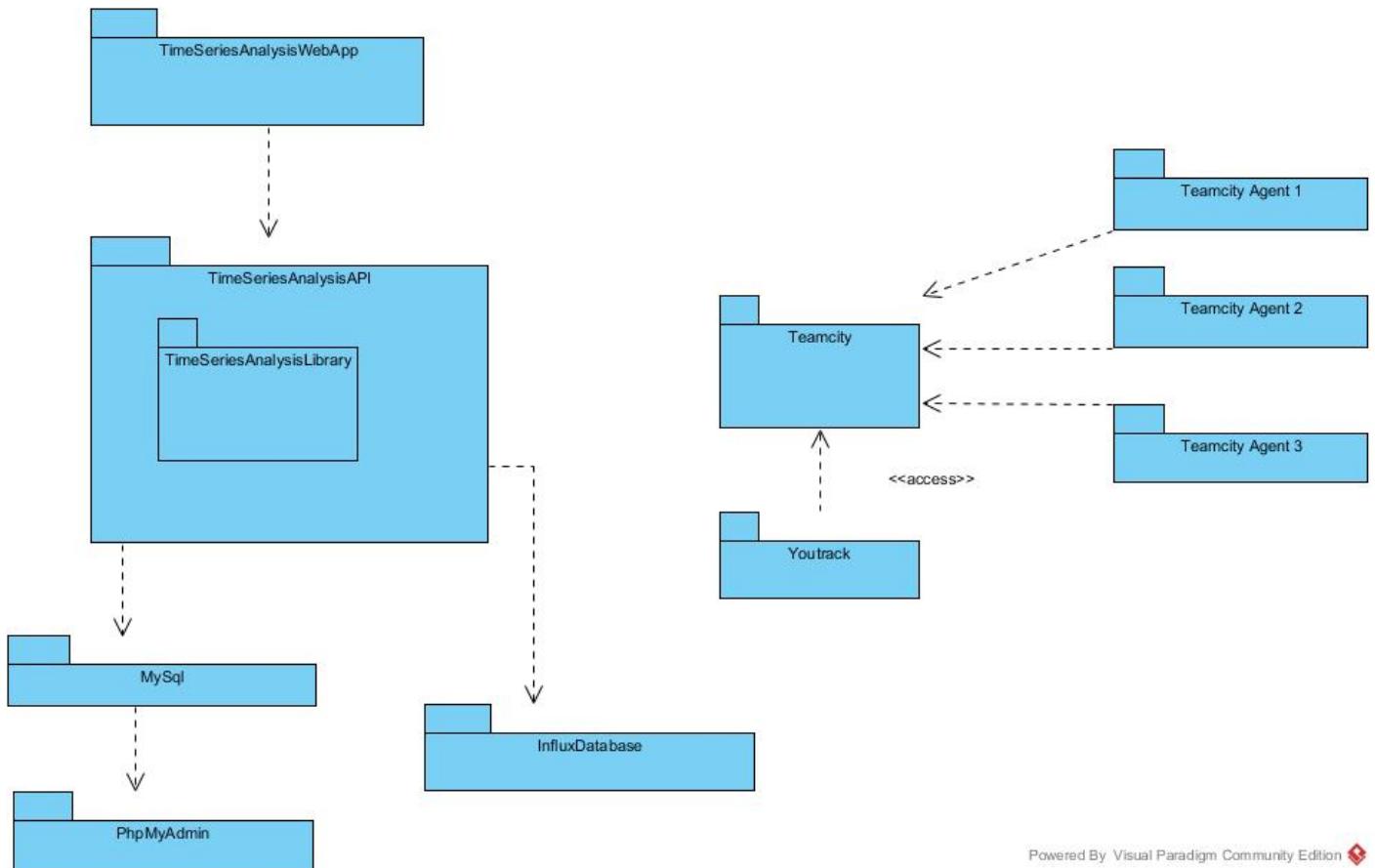


Imagen 21: Diagrama de contenedores Docker.

Powered By Visual Paradigm Community Edition

## 4.2. Biblioteca de análisis en R (TimeSeriesAnalysisLibrary)

La biblioteca de análisis desarrollada en R y C++ es el módulo básico para el análisis de series temporales, todo cálculo que se hace con la serie temporal se ejecuta en la biblioteca obteniendo un resultado.

La biblioteca ha sido desarrollada siguiendo la programación orientada a objetos. El desarrollo inicio con las clases para poder gestionar series temporales, cargándolas desde archivos, pudiendo exportarlas, y hacer cálculos sobre ellas. Se han diseñado clases contenedoras como TimeSeries, TimeSeriesData o para colecciones de datos "Helpers" para hacer cálculos más simples. Todo esto haciendo uso de un gestor de excepciones para mostrar cualquier tipo de error al usuario. Además se crearon clases abstractas sobre las que se crearían las clases finales, ejemplos de esto son las clases Method, Result o Collection. Todas estas clases se pueden ver en el diagrama mostrado anteriormente.

Debido a problemas de eficiencia en algunos casos, por ejemplo el cálculo del número de centros con el método Chiu, se tuvieron que desarrollar algunas funciones en C++, y el cambio fue muy notable; en algunos casos se pasó de horas a segundos.

Todas las clases desarrolladas en R han sido testeadas para evitar fallos en modificaciones posteriores. Esto se hizo debido a que esta biblioteca es la base de todo el sistema superior y debía funcionar sin problemas.

Estas clases pertenecen a diversos paquetes que se han desarrollado y que se explicaran a continuación, y todos los paquetes juntos forman la biblioteca de análisis TimeSeriesAnalysisLibrary, la cuál ha sido instalada en el servidor junto a todos los paquetes de los que depende. Y se accede mediante scripts escritos en R que son los que llaman los archivos en PHP pertenecientes a la API.

Los paquetes R desarrollados han sido los siguientes:

- RLog: Ofrece funcionalidad para gestionar logs de la aplicación y algunas excepciones.
- DataUtils: Clases para trabajar con colecciones de datos de cualquier

tipo, además de colecciones de un tamaño muy grande y que son necesarias de almacenar en disco.

- Clustering: Clases para ejecutar métodos de Clustering sobre conjuntos de datos.
- TimeSeries: Clases para trabajar con series temporales.
- TimeSeriesComplexity: Métodos para calcular medidas de complejidad a series temporales.
- TimeSeriesForecast: Métodos para aplicar métodos de predicción a series temporales.
- TimeSeriesTransformation: Métodos para aplicar métodos de transformación a series temporales.
- TimeSeriesDatabase: Clases para acceder a TimeSeriesAnalysisAPI.
- TimeSeriesAnalysis: Paquete formado por los paquetes anteriores y que contiene scripts para ejecutar análisis de series temporales.

Este módulo es el sistema de clasificación, se usa mediante scripts R que se han desarrollado. El paquete TimeSeriesAnalysis es el que se usa en todos los scripts, ya que aglutina el resto de paquetes. El sistema de clasificación funciona a partir de cuatro scripts: UploadTimeSeries, ApplyComplexityMeasures, Clustering y Forecasting.

- UploadTimeSeries: Este script se utiliza para subir grandes cantidades de series temporales almacenadas en un directorio, realiza llamadas Http POST sobre TimeSeriesAnalysisAPI.
- ApplyComplexityMeasures: Este script aplica todas las medidas de complejidad sobre todas las series temporales almacenadas en la base de datos y guarda los resultados en la base de datos.
- Clustering: Obtiene los resultados de medidas de complejidad almacenados en la base de datos y aplica métodos de clustering generando todos los centros, y guardando estos también en la base de datos junto a sus coordenadas.

- Forecasting: Aplica métodos de predicción sobre las series temporales de cada grupo de clustering y selecciona el método de menor error, también almacena los resultados en la base de datos. El script Clustering y Forecasting funcionan de forma conjunta.

Usando estos scripts se obtiene el sistema de clasificación a falta de la última parte que es la que hace uso la aplicación web: clasificar una nueva serie temporal. Cuando se intenta clasificar una serie temporal la aplicación web llama a la API y esta aplica todas las medidas de complejidad a la serie, la coloca en un clustering calculando su distancia a los clusters y seleccionando el de menor; y calcula su predicción con el método que se selecciono para el cluster.

### **4.3. API PHP (TimeSeriesAnalysisAPI)**

Esta sección trata sobre TimeSeriesAnalysisAPI, este es el módulo al que se hacen las peticiones REST desde la aplicación web y que hace las llamadas a la base de datos y a TimeSeriesAnalysisLibrary. Además de la biblioteca de métodos en R este módulo también se ha desarrollado siguiendo la programación orientada a objetos.

Este módulo contiene la siguiente funcionalidad:

- Recibir y responder a las peticiones HTTP.
- Acceder a la base de datos SQL.
- Acceder a la base de datos InfluxDB.
- Hacer la llamada a TimeSeriesAnalysisLibrary.

La base de todo el módulo es poder gestionar series temporales y usuarios, que son el modelo del sistema. Por lo tanto se han implementado clases que las gestionan, ya sea su creación, modificación y eliminación. También eran necesarios métodos de validación para el registro e inicio de sesión de usuarios y han sido implementados en esta API.

Además hacía falta poder gestionar el inicio de sesión y el subir archivos de series temporales al servidor. Todas estas clases necesarias fueron las primeras en desarrollarse.

Para recibir y responder a las peticiones HTTP ha sido usado el framework Slim ya que genera una API REST de manera muy sencilla, estos archivos son a los que se han llamado controladores.

El acceso a InfluxDB se hace mediante un paquete que nos ofrece el mismo software para PHP. Y para las llamadas a TimeSeriesAnalysisLibrary se hace uso del paquete para PHP: kachkaev/php-r [23].

El problema de este paquete es que los resultados no se devuelven correctamente formateados así que hubo que hacer un parseador para R en PHP.

#### 4.4. Angular aplicación web (TimeSeriesAnalysisApp)

El modulo superior de todo el desarrollo corresponde a la aplicación web TimeSeriesAnalysisApp. Ha sido desarrollada usando el framework de Google: Angular. Este framework usa Typescript para generar la aplicación, y estos archivos son compilados a Javascript.

La aplicación web está formada por:

- Una página principal que explica un poco la funcionalidad de la aplicación y muestra un botón para iniciar la aplicación de análisis.
- Una página que muestra una lista de las series temporales públicas y la opción de subir nuevas series temporales, además de iniciar sesión o registrarse.
- Una página que muestra cada serie temporal con sus propiedades, una gráfica y los posibles calculos que se pueden hacer sobre ella.
- Una página de perfil de usuario donde muestra los datos del usuario y sus series temporales ya sean públicas o privadas.

Además de la implementación de estas páginas ha sido necesaria la creación de servicios para acceder a la API, y diversos componentes para el diseño de la vista (Modales, ...).

Para una mayor eficiencia todos los archivos TypeScript, que son compilados a JavaScript, han sido posteriormente agrupados en un solo archivo y minificado.

En cuanto a las vistas, es decir los archivos HTML, en un primer momento se

creaban por separado del archivo TypeScript, pero para encapsular la aplicación entera en un solo archivo se han definido estos archivos como TypeScript y así son comprimidos dentro del mismo archivo, consiguiendo la aplicación completa (a excepción de imágenes y estilos) en un solo archivo.

Uno de los puntos más importantes de la implementación de esta aplicación web, a excepción del análisis en R pero esto al fin y al cabo consistía en una llamada a la API, han sido las validaciones in-line en todos los formularios; ya que no se pensaron en un principio y hubo que modificar la API y los formularios para mostrar bien los posibles errores.

## 4.5. Base de datos MySQL

Fue necesaria la construcción de una base de datos en MySQL para gestionar usuarios, series temporales, y los resultados de medidas de complejidad y clustering. El diagrama de la base de datos se puede ver en la Imagen 2.

La base de datos cuenta con cuatro entidades: users, timeseries, complexity-measures, clustering. La entidad users es necesaria para los usuarios de la aplicación web, se ha considerado una gestión de usuarios para poder subir series temporales.

La entidad timeseries se ha considerado para poder obtener almacenar las series temporales, sin valores pero si con datos necesarios (número de valores, usuario que la subió).

Por último complexitymeasures y clustering son necesarias para el sistema de clasificación, la primera almacena los resultados de las medidas de complejidad de todas las series temporales y la segunda los resultados de clustering sobre todas las medidas de complejidad (de la entidad anterior) con los métodos de predicción adecuados para cada centro.

## 5. Conclusiones y vías futuras

El proyecto realizado se compone de tres módulos principales: una biblioteca formada con los métodos de análisis necesarios (TimeSeriesAnalysisLibrary), una base de datos de series temporales, una API (TimeSeriesAnalysisAPI) de acceso a datos y métodos; y una interfaz gráfica de usuario (en web) para su uso interactivo (TimeSeriesAnalysisWebApp).

El módulo TimeSeriesAnalysisLibrary está formada por un conjunto de paquetes en R que ofrecen variada funcionalidad, como son clases para trabajar con grandes cantidades de datos, métodos para acceder a la base de datos haciendo llamadas a la API e incluso opciones para trabajo con núcleos en paralelo.

En este módulo se incluye el objetivo principal de este proyecto que es el desarrollo de una biblioteca de análisis que incluye un sistema de clasificación. La base de este sistema de clasificación son las series temporales y más concretamente sus resultados de complejidad, los cuales hablaremos más adelante, y la funcionalidad que se quiere obtener con esto es clasificar series temporales y dependiendo de su tipo aplicar un método de predicción automáticamente. Su funcionamiento es:

- Usa como conocimiento toda la base de datos de series temporales.
- Calcula la complejidad de cada serie temporal y la almacena.
- Ejecuta métodos de clustering para obtener un modelo de clasificación de series temporales.
- Sobre cada serie temporal de cada grupo del clustering (cluster) se calculan predicciones con varios métodos de predicción.
- De cada cluster se selecciona el método de predicción con error medio menor.
- Para calcular el cluster de una nueva serie temporal (clasificación) se usa la complejidad de la misma y la distancia a todos los clusters para seleccionar el de menor distancia.
- Y usando el método de menor error de este cluster se obtiene la predicción automática.

En otro nivel está el modulo TimeSeriesAnalysisAPI el cuál gestiona todas las llamadas a la base de datos y a la biblioteca de análisis.

La aplicación web (TimeSeriesAnalysisWebApp) se puede decir que es la parte visible del proyecto, tiene como datos de entrada una serie temporal en un formato de archivo determinado, además de determinadas variables que la definen, que son introducidas por el usuario. Las series temporales introducidas serán una lista, o varias si es multivariada, siendo cada elemento un valor en determinado momento temporal. Estos valores serán leídos correctamente, formateados para poder trabajar con ellos y almacenados en la base de datos. La vista principal de una serie temporal en la aplicación es una visión general de las propiedades de la serie temporal, una gráfica de los datos de esta, y un modo de interacción para poder aplicarle métodos. Los métodos que se pueden aplicar a las series temporales son medidas de complejidad, transformaciones, métodos de predicción y de clasificación.

La interfaz de la aplicación cuando se inicia es una página con datos sobre series temporales y análisis de estas. Cuando se accede a la aplicación de análisis se observa una página formada por un apartado donde subir un archivo y una lista de las series temporales disponibles con las que trabajar, y cuando se selecciona una serie temporal podemos acceder a un apartado donde trabajar con ella.

La aplicación se ha dividido en una arquitectura de tres niveles.

- Back-end: Nivel formado por el servidor Apache, la base de datos SQL, la base de datos de series temporales InfluxDB [14] y la biblioteca de métodos de análisis en R [48], esto último es el módulo que se ha llamado TimeSeriesAnalysisLibrary, y está formado por un conjunto de paquetes R desarrollados.
- Middleware: Nivel formado por la API que hace de intermediaria entre los datos (back-end) y la aplicación web, desarrollada haciendo uso de la arquitectura REST con PHP [49], a este módulo se ha llamado TimeSeriesAnalysisAPI.
- Front-end: La aplicación web desarrollada con el framework Angular [8], y por último este módulo es TimeSeriesAnalysisApp.

Gracias al desarrollo de la API toda la parte de computación se elimina de la aplicación web y se deja en esta la única responsabilidad de la vista de

usuario y las peticiones a la API.

Debido a que para el desarrollo del sistema se ha seguido una metodología ágil, específicamente Scrum [50], obtenemos unos costes mínimos frente a los cambios de requisitos, además de una gran facilidad de añadir nuevos módulos al sistema o características.

Los objetivos que se han conseguido han sido los siguientes:

- Una biblioteca de análisis formada por varios paquetes en R necesarios para trabajar con todo tipo de series temporales y devolver resultados.
- Un modelado de datos que permite almacenar las series temporales haciendo uso de una base de datos InfluxDB para almacenar las series temporales completas y otra SQL donde se almacenan las características principales de estas.
- Una API (Application Programming Interface) que accede tanto a la biblioteca de análisis como a las bases de datos.
- Una aplicación web robusta, ligera, fácil de usar y con un gran diseño que ofrece toda la funcionalidad necesaria para trabajar con series temporales: subir una nueva serie temporal, mostrar series temporales almacenadas (sus datos y sus características), aplicar métodos de análisis y observar los resultados. Además la aplicación cuenta con un apartado donde obtener información sobre series temporales, los métodos de análisis implementados, la aplicación web en general, etc.

Se puede concluir que se han conseguido los objetivos planificados al inicio del proyecto: Construir un sistema basado en el conocimiento, ofrecer a usuarios una aplicación web desde la que analizar una serie temporal y una biblioteca de análisis de series temporales con mucha funcionalidad.

El primer y último objetivo se demuestra su realización en la sección 3 y en la sección 4.2. El segundo objetivo se demuestra en la sección 4.4.

## 6. Apéndices

### 6.1. Apéndice A: Bibliografía

### Referencias

- [1] S. Abe. *Neural Networks and Fuzzy Systems: Theory and Applications*. Springer US, 2012.
- [2] Juan Carlos Abril. Structural time series models, 2014. [Internet; descargado 26-mayo-2017].
- [3] Bootstrap. Bootstrap, 2017. [Internet; descargado 19-junio-2017].
- [4] Winston Chang. R6, 2017. [Internet; descargado 19-junio-2017].
- [5] Composer. Composer, 2017. [Internet; descargado 19-junio-2017].
- [6] Nikolic-Doric D.T.Mihailovic, .Mimic and I.Arsebic. Novel measures based on the kolmogorov complexity for use in complex system behavior studies and time series analysis. <http://arxiv.org/pdf/1310.1304v1.pdf>. [Internet; descargado 24-Mayo-2017].
- [7] Dirk Eddelbuettel. Rcpp, 2017. [Internet; descargado 19-junio-2017].
- [8] Google. Angular google. <https://angular.io/>. [Internet; descargado 26-Octubre-2016].
- [9] Guzzle. Guzzle http, 2017. [Internet; descargado 19-junio-2017].
- [10] Jean Hausser and Korbinian Strimmer. Entropy, 2017. [Internet; descargado 19-junio-2017].
- [11] Jean Hausser and Korbinian Strimmer. Package ‘entropy’. <https://cran.r-project.org/web/packages/entropy/entropy.pdf>, 2017. [Internet; descargado 26-Mayo-2017].
- [12] Highcharts. Highcharts, 2017. [Internet; descargado 19-junio-2017].
- [13] Rob Hyndman. Forecast, 2017. [Internet; descargado 19-junio-2017].
- [14] InfluxDB. Influxdb. <https://www.influxdata.com/>. [Internet; descargado 26-Octubre-2016].

- [15] Influxdb-php. Influxdb-php, 2017. [Internet; descargado 19-junio-2017].
- [16] Joshua M. Ulrich Jeffrey A. Ryan. Xts, 2017. [Internet; descargado 19-junio-2017].
- [17] Jetbrains. PhpStorm ide, 2017. [Internet; descargado 19-junio-2017].
- [18] Jetbrains. Pycharm ide, 2017. [Internet; descargado 19-junio-2017].
- [19] Jetbrains. Teamcity, 2017. [Internet; descargado 19-junio-2017].
- [20] Jetbrains. Youtrack, 2017. [Internet; descargado 19-junio-2017].
- [21] Rob Jhyndman. forecast.stl function. [Internet; descargado 26-mayo-2017].
- [22] Rob Jhyndman. Forecasting time series with complex seasonal patterns using exponential smoothing, 2010. [Internet; descargado 26-mayo-2017].
- [23] kachkaev. Php-r. <https://github.com/kachkaev/php-r>. [Internet; descargado 22-Noviembre-2016].
- [24] Michael J. Kane. Bigmemory, 2017. [Internet; descargado 19-junio-2017].
- [25] Karma. Karma, 2017. [Internet; descargado 19-junio-2017].
- [26] Da-Guan Ke and Qin-Ye Tong. Easily adaptable complexity measure for finite time series. <http://arxiv.org/pdf/nlin/0505052.pdf>. [Internet; descargado 24-Mayo-2017].
- [27] Teja Kodali. Hierarchical clustering in r, 2017. [Internet; descargado 26-mayo-2017].
- [28] M. G. Cosenza L. A. Molina, M. Escalona-Morán. Complejidad estadística en series temporales. [http://www.ciens.ula.ve/cff/caoticos/PDFs/EscalonaMolinaCosenza2008\\_NAC.pdf](http://www.ciens.ula.ve/cff/caoticos/PDFs/EscalonaMolinaCosenza2008_NAC.pdf). Internet; descargado 1-Noviembre-2016].
- [29] Ngx-modal. Ngx-modal, 2017. [Internet; descargado 19-junio-2017].
- [30] Sebastian Nowozin. Estimating discrete entropy, part 2. <http://www.nowozin.net/sebastian/blog/estimating-discrete-entropy-part-2.html>, 2017. [Internet; descargado 26-Mayo-2017].

- [31] Visual Paradigm. Visual paradigm, 2017. [Internet; descargado 19-junio-2017].
- [32] Phpunit. Phpunit, 2017. [Internet; descargado 19-junio-2017].
- [33] R-core. parallel, 2017. [Internet; descargado 19-junio-2017].
- [34] RcppParallel. Rcppparallel, 2017. [Internet; descargado 19-junio-2017].
- [35] Rodrigo Paredes Ricardo Baeza Yates. Complejidad de kolmogorov. <https://users.dcc.uchile.cl/~raparede/clases/cc40a/kolmogorov/clase.pdf>. [Internet; descargado 24-Mayo-2017].
- [36] Slim. Slim framework, 2017. [Internet; descargado 19-junio-2017].
- [37] sthda. Model-based clustering, 2017. [Internet; descargado 26-mayo-2017].
- [38] Korbinian Strimmer. entropy: Chao-shen. <https://rdrr.io/cran/entropy/man/entropy.ChaoShen.html>, 2017. [Internet; descargado 26-Mayo-2017].
- [39] Korbinian Strimmer. entropy: Dirichlet. <https://www.rdocumentation.org/packages/entropy/versions/1.2.1/topics/entropy.Dirichlet>, 2017. [Internet; descargado 26-Mayo-2017].
- [40] Korbinian Strimmer. entropy.shrink function. <https://www.rdocumentation.org/packages/entropy/versions/1.2.1/topics/entropy.shrink>, 2017. [Internet; descargado 26-Mayo-2017].
- [41] Korbinian Strimmer. R: Miller-madow. <https://artax.karlin.mff.cuni.cz/r-help/library/entropy/html/entropy.MillerMadow.html>, 2017. [Internet; descargado 26-Mayo-2017].
- [42] ueltschi. Shannon entropy. <http://www.ueltschi.org/teaching/chapShannon.pdf>, 2017. [Internet; descargado 26-Mayo-2017].
- [43] Hadley Wickham. testthat, 2017. [Internet; descargado 19-junio-2017].
- [44] Wikipedia. Serie temporal — wikipedia, la enciclopedia libre. [https://es.wikipedia.org/w/index.php?title=Serie\\_temporal&oldid=86168679](https://es.wikipedia.org/w/index.php?title=Serie_temporal&oldid=86168679), 2015. [Internet; descargado 25-octubre-2016].

- [45] Wikipedia. Modelo autorregresivo integrado de media móvil — wikipedia, la enciclopedia libre, 2016. [Internet; descargado 26-mayo-2017].
- [46] Wikipedia. Modelo-vista-controlador — wikipedia, la enciclopedia libre. <https://es.wikipedia.org/w/index.php?title=Modelo%E2%80%93vista%E2%80%93controlador&oldid=94314709>, 2016. [Internet; descargado 27-octubre-2016].
- [47] Wikipedia. Npm — wikipedia, la enciclopedia libre, 2016. [Internet; descargado 19-junio-2017].
- [48] Wikipedia. R (lenguaje de programación) — wikipedia, la enciclopedia libre. [https://es.wikipedia.org/w/index.php?title=R\\_\(lenguaje\\_de\\_programaci%C3%B3n\)&oldid=93673065](https://es.wikipedia.org/w/index.php?title=R_(lenguaje_de_programaci%C3%B3n)&oldid=93673065), 2016. [Internet; descargado 26-octubre-2016].
- [49] Wikipedia. Representational state transfer — wikipedia, la enciclopedia libre. [https://es.wikipedia.org/w/index.php?title=Representational\\_State\\_Transfer&oldid=93410210](https://es.wikipedia.org/w/index.php?title=Representational_State_Transfer&oldid=93410210), 2016. [Internet; descargado 27-octubre-2016].
- [50] Wikipedia. Scrum (desarrollo de software) — wikipedia, la enciclopedia libre. [https://es.wikipedia.org/w/index.php?title=Scrum\\_\(desarrollo\\_de\\_software\)&oldid=94110496](https://es.wikipedia.org/w/index.php?title=Scrum_(desarrollo_de_software)&oldid=94110496), 2016. [Internet; descargado 27-octubre-2016].
- [51] Wikipedia. Suavizamiento exponencial — wikipedia, la enciclopedia libre, 2016. [Internet; descargado 26-mayo-2017].
- [52] Wikipedia. Sucesión de fibonacci — wikipedia, la enciclopedia libre. [https://es.wikipedia.org/w/index.php?title=Sucesi%C3%B3n\\_de\\_Fibonacci&oldid=94573974](https://es.wikipedia.org/w/index.php?title=Sucesi%C3%B3n_de_Fibonacci&oldid=94573974), 2016. [Internet; descargado 27-octubre-2016].
- [53] Wikipedia. Agrupamiento jerárquico — wikipedia, la enciclopedia libre, 2017. [Internet; descargado 26-mayo-2017].
- [54] Wikipedia. Algoritmo de agrupamiento — wikipedia, la enciclopedia libre, 2017. [Internet; descargado 26-mayo-2017].

- [55] Wikipedia. Claude elwood shannon — wikipedia, la enciclopedia libre. [https://es.wikipedia.org/w/index.php?title=Claude\\_Elwood\\_Shannon&oldid=98862064](https://es.wikipedia.org/w/index.php?title=Claude_Elwood_Shannon&oldid=98862064), 2017. [Internet; descargado 26-mayo-2017].
- [56] Wikipedia. Docker (software) — wikipedia, la enciclopedia libre, 2017. [Internet; descargado 19-junio-2017].
- [57] Wikipedia. Fuzzy clustering — wikipedia, la enciclopedia libre, 2017. [Internet; descargado 26-mayo-2017].
- [58] Wikipedia. Git — wikipedia, la enciclopedia libre, 2017. [Internet; descargado 19-junio-2017].
- [59] Wikipedia. K-means — wikipedia, la enciclopedia libre, 2017. [Internet; descargado 26-mayo-2017].
- [60] Wikipedia. Kolmogorov complexity — wikipedia, the free encyclopedia. [http://en.wikipedia.org/w/index.php?title=Kolmogorov\\_complexity&oldid=646881276](http://en.wikipedia.org/w/index.php?title=Kolmogorov_complexity&oldid=646881276), 2017. [Internet; descargado 24-Mayo-2017].
- [61] Wikipedia. Máquina de turing — wikipedia, la enciclopedia libre. [https://es.wikipedia.org/w/index.php?title=M%C3%A1quina\\_de\\_Turing&oldid=98707795](https://es.wikipedia.org/w/index.php?title=M%C3%A1quina_de_Turing&oldid=98707795), 2017. [Internet; descargado 24-mayo-2017].
- [62] Wikipedia. Phpmyadmin — wikipedia, la enciclopedia libre, 2017. [Internet; descargado 19-junio-2017].
- [63] Wikipedia. Rstudio — wikipedia, la enciclopedia libre, 2017. [Internet; descargado 19-junio-2017].
- [64] Wikipedia. Servidor http apache — wikipedia, la enciclopedia libre, 2017. [Internet; descargado 19-junio-2017].
- [65] Wiktionary. Shannon entropy — wiktionary, the free dictionary. [https://en.wiktionary.org/w/index.php?title=Shannon\\_entropy&oldid=45005014](https://en.wiktionary.org/w/index.php?title=Shannon_entropy&oldid=45005014), 2017. [Internet; descargado 26-Mayo-2017].

## 6.2. Apéndice B: Glosario de Términos

- **Serie temporal:** Una serie temporal o cronológica es una secuencia de datos, observaciones o valores, medidos en determinados momentos y ordenados cronológicamente.
- **Análisis de series temporales:** Estudio de las propiedades de una serie temporal. Para el análisis de las series temporales se usan métodos que ayudan a interpretarlas y que permiten extraer información representativa sobre las relaciones subyacentes entre los datos de la serie o de diversas series.
- **Predicción de series temporales:** Averiguar valores desconocidos en un determinado momento de una serie temporal. El análisis de una serie temporal permite en diferente medida y con distinta confianza extrapolación o interpolar los datos y así predecir el comportamiento de la serie en momentos no observados, sean en el futuro (extrapolación pronóstica), en el pasado (extrapolación retrógrada) o en momentos intermedios (interpolación).
- **Medida de complejidad:** Proceso que consigue recibiendo un conjunto de datos, serie temporal, devolver un número que define la complejidad de los datos.
- **Formato:** Un formato de archivo es un estándar que define la manera en que está codificada la información en un archivo.
- **Propiedades principales de una serie temporal:** Valores medidos, fechas en las que se midió cada valor, número de valores medidos por cada fecha (series temporales multivariadas).
- **Frecuencia de una serie temporal:** Frecuencia con la que se mide un nuevo valor.
- **Clustering:** Es un procedimiento de agrupación de una serie de vectores de acuerdo con un criterio. Generalmente, los vectores de un mismo grupo (o clústers) comparten propiedades comunes.
- **Centro de un cluster:** En un clustering con un algoritmo de agrupamiento no jerárquico, el número de grupos se determina de antemano y las observaciones se van asignando a los grupos en función de su cercanía.
- **Log:** Un log es un registro oficial de eventos durante un rango de tiempo en particular