UNIVERSIDAD DE CASTILLA-LA MANCHA

ESCUELA SUPERIOR DE INFORMÁTICA

ANTEPROYECTO DEL TRABAJO FIN DE GRADO

GRADO EN INGENIERÍA INFORMÁTICA

INGENIERÍA DEL SOFTWARE

Uso de tecnología CEP para la detección de desgaste en aerogeneradores, “Eolic Event Consumer”

Using CEP to detect exhaustion in wind turbines, “Eolic Event Consumer”

Autor: Enrique Brazález Segovia.
Director: Gregorio Diaz Descalzo.

Febrero, 2018.

Índice de Contenido

1. Introducción. 1
1.1. Motivación. 1
1.2. Descripción del proyecto. 2
2. Tecnología específica / Intensificación / Itinerario cursado por el alumno. 3
3. Objetivos. 6
4. Método y fases de desarrollo. 6
5. Medios que se pretenden utilizar. 8
5.1. Medios Hardware. 8
5.2. Medios Software. 8
Referencias. 10

Índice de Ilustraciones

Ilustración 1. Esquema de arquitectura. 2
Ilustración 2. Diagrama de Gantt de planificación de tiempo. 8

Índice de Tablas

Tabla 1. Tecnología Específica cursada por el alumno. 3
Tabla 2. Justificación de las competencias específicas abordadas en el TFG. 3

1. Introducción.

"Internet of Things" (IoT), o "Internet de las cosas" en español, es el tema que ha inspirado este proyecto. Cómo poder comunicar todos los elementos que influyen en nuestro día a día puede parecer poco viable, debido a la gran inversión y trabajo que se necesita. No obstante, cada día que pasa dicho esfuerzo es menor [1]. Así, la evolución del IoT cambiará nuestro futuro. Nuestra forma de percibir la realidad cambiará totalmente y no sólo en nuestra vida cotidiana. si no que irá mucho más allá afectando a ámbitos como la educación, la comunicación, las empresas, la ciencia, y el gobierno. En este trabajo de fin de grado nos centraremos en el ámbito científico e industrial, en concreto el que afecta a las energías renovables.

1.1. Motivación.

Este proyecto nace de la necesidad de optimizar la obtención de energía a un bajo coste y teniendo en cuenta la preservación del medio ambiente. Las fuentes de energía no renovables son aquellas que se encuentran de forma limitada en nuestro planeta dado que su consumo es mayor que su "regeneración" [2].

La integración de los sistemas de información en el sector industrial está al orden del día, es decir, la industria está vitalmente unida a los sistemas software en su enorme mayoría. Si la tecnología decae, cualquier actividad profesional relacionada está destinada al fracaso como es el caso de la generación de energía [3]. Existen multitud de aplicaciones software para controlar las herramientas de extracción de energía por medios renovables. Por ejemplo, en placas solares móviles podemos observar un sistema para ajustar la orientación con respecto al sol y optimizar su producción y, por otro lado, en el sistema de control de capacidad en un embalse podemos decidir cuándo nos conviene aprovechar la energía producida por su caudal.

A lo que energía renovable se refiere, hemos de destacar la energía eólica y la implicación que tienen los aerogeneradores en la misma. La integración del software y sistemas electrónicos en todos sus componentes tanto a nivel individual, así como, a nivel de parque eólico es abrumador. Un ejemplo destacable sería el flujo de información que se produce entre los aerogeneradores y las estaciones centrales. Sin embargo, todo ese volumen de información no se aprovecha nada más que para labores muy sencillas de mantenimiento y monitorización.

El objetivo de este proyecto es ir un paso más allá y en base a este gran volumen de información predecir el desgaste de un aerogenerador. Como dijo Bill Gates: "La información es poder". Llevando esta frase a la práctica. Si tenemos toda la infraestructura de sensorización, ¿por qué no aprovechar esos grandes volúmenes de datos no solamente para labores de control y monitorización sino también para realizar predicciones?

1.2. Descripción del proyecto.

Dando respuesta a la pregunta lanzada anteriormente surge como posible solución el Procesamiento de Eventos Complejos (tecnología CEP) como aplicación de "Fast Data". La mejor forma de tomar decisiones en estos ámbitos es hacerlo cuanto antes, de ahí que se examine toda la información a tiempo real para de esta forma dirigir satisfactoriamente la toma de decisiones o acciones correctivas en el ámbito industrial. Hay que saber distinguir que se habla de "Fast Data" y no de "Big Data" [4]. ¿Por qué? Fast Data tiene una ventaja, y es que no almacena ningún tipo de información. Conforme los datos son generados, una vez procesados por primera vez, se desechan, es decir, se analizan conforme van siendo generados para después desecharlos, lo que lo hace muy rentable. Si se almacenara hablaríamos de "Big Data". Por lo tanto, como se genera, se procesa, y se desecha sin almacenarla hablamos de "Fast Data", evitándonos el gran coste que llevaría salvaguardar tanta información.

La ventaja del procesamiento de eventos complejos tal y como Fast Data determina, es que puede dominar cualquier volumen de datos, a una velocidad en tiempo real y de orígenes muy diversos. CEP, es una tecnología emergente capaz de monitorizar múltiples flujos de eventos, procesarlos mediante la determinación de patrones implementados en Esper Event Processing Language (EPL) [5], y así tomar decisiones tanto predictivas como correctivas en base a la información que el contexto nos aporta.

Como conclusión a la problemática generada por la necesidad de optimizar la producción en un parque eólico, este trabajo de fin de grado tiene como meta la predicción del desgaste de aerogeneradores a través de un sistema SOA 2.0. SOA 2.0, es una arquitectura orientada a servicios, aunque dirigida por eventos [6]. Un esquema de "Eolic Event Consumer", para solucionar el caso de estudio, podremos verlo en la arquitectura propuesta en la Ilustración 1.

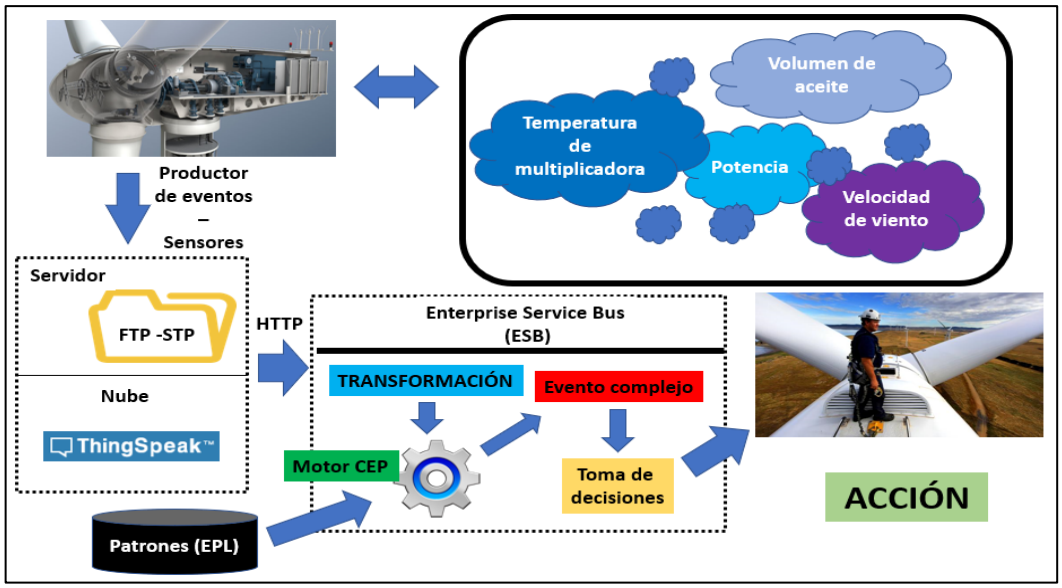

Ilustración 1. Esquema de arquitectura.

Como vemos, el origen de los datos proviene de los múltiples sensores fluyendo vía HTTP para transmitirlos al motor CEP. El motor CEP, procesa la información utilizando los patrones implementados. De esta forma, desplegaremos los eventos complejos en una aplicación gráfica aplicando medidas correctivas y preventivas dependiendo del evento complejo que se detecte en el flujo de datos.

2. Tecnología específica / Intensificación / Itinerario cursado por el alumno.

La intensificación que toca directamente este proyecto de fin de grado es la de Ingeniería del Software. Es una rama que estudia el entendimiento y análisis del problema, planificación, modelado y diseño del sistema software solución, implementación del código y la evaluación de la exactitud del resultado [7].

Tabla 1. Tecnología Específica cursada por el alumno.

	Marcar la tecnología cursada
	Tecnologías de la Información
	Computación
	Ingeniería del Software
	Ingeniería de Computadores

La enumeración de competencias y justificación para su satisfacción queda redactada en la Tabla 2, razonando lo exigido en el anexo de la convocatoria.

Tabla 2. Justificación de las competencias específicas abordadas en el TFG.

Competencias	**Justificación**
Competencia 1. Capacidad para desarrollar, mantener y evaluar servicios y sistemas software que satisfagan todos los requisitos del usuario y se comporten de forma fiable y eficiente, sean asequibles de desarrollar y mantener y cumplan normas de calidad, aplicando las teorías, principios, métodos y	Llevaremos a cabo este proyecto a través de una metodología ágil, de forma que cuidaremos todos los aspectos referentes a la recogida de requisitos (técnica de storyboards), su desarrollo (con control de versiones), y evaluación del producto (utilizando pruebas unitarias, de integración y aceptación) puesto que mantendremos contacto continuo con el Product Owner. De esta forma nos aseguraremos de que el software finalmente cumple con lo impuesto. Usaremos normas de calidad, utilizando

prácticas de la Ingeniería del Software.	un modelo de calidad como es el defendido e impuesto en la ISO-IEC 25010 [8]. Por lo que cumpliremos y defenderemos esta primera competencia.
Competencia 2. Capacidad para valorar las necesidades del cliente y especificar los requisitos software para satisfacer estas necesidades, reconciliando objetivos en conflicto mediante la búsqueda de compromisos aceptables dentro de las limitaciones derivadas del coste, del tiempo, de la existencia de sistemas ya desarrollados y de las propias organizaciones.	Al usar una metodología ágil para elaborar este trabajo de fin de grado, estaremos continuamente en contacto con el cliente, que nos ayudará a dar el visto bueno a los requisitos y objetivos marcados. Además, se llevará a cabo una elicitación y recogida de requisitos para analizar cuáles son los objetivos que la aplicación debe de alcanzar y con qué criterio se darán por satisfechos. Para conciliar y amarrar esas necesidades se fijará un contrato que comprometa el desarrollo e implementación de estas funcionalidades, para que de esta forma se fijen unas condiciones que garanticen su entrega en tiempo y coste asequible. Asegurándonos así, que se cumplan tanto los intereses del cliente, como los nuestros como desarrolladores, y que conforme avance el proyecto no se pida continuamente funcionalidades nuevas. De esta forma, se cumpliría con esta competencia, y se fijaría el compromiso tanto de la empresa con el desarrollador, y el desarrollador con el proyecto.
Competencia 3. Capacidad de dar solución a problemas de integración en función de las estrategias, estándares y tecnologías disponibles.	Los problemas de integración serán resueltos debido a que usaremos herramientas como: - **Team Foundation Service.** Para coordinar todas y cada una de las tareas que serán necesarias para desarrollar el proyecto. Aportando así un control de errores, riesgos posibles que se pueden dar, control del trabajo a través del Sprint Burndown, etc. - **ESB.** Al usar el Enterprise Service Bus de Mule, nos permitirá integrar multitud de tecnologías como, por ejemplo, la comunicación entre un servicio FTP que alimente el motor CEP para detectar los eventos de interés. Para que

	posteriormente se mande por servicio SMTP de correo electrónico, la alerta de la ocurrencia de los eventos detectados por los patrones. - **GitHub.** Para hacer un control de versiones y tener unificado todos los aspectos del código.
Competencia 4. Capacidad de identificar y analizar problemas y diseñar, desarrollar, implementar, verificar y documentar soluciones software sobre la base de un conocimiento adecuado de las teorías, modelos y técnicas actuales.	Esta competencia se dará por satisfecha puesto que, en uno de los capítulos de la memoria, el estado del arte, veremos enfrentadas la discusión entre las posibles soluciones a nuestro problema a resolver. Además, veremos un debate de cuáles son todas las soluciones posibles, y por qué hemos decidido que la nuestra es la más viable. Como aditivo, también se verán todos los conocimientos teóricos y técnicos referentes tanto al software implementado, como a las bases de energía eólica de las que nacen esos eventos frutos del desgaste de los aerogeneradores. Aplicando todos los conocimientos expuestos en este capítulo pasaremos al desarrollo ágil de la aplicación con técnicas aprendidas a lo largo de la carrera y haciendo equilibrio entre técnicas más actuales como son el empleo de storyboards, y otras más tradicionales como son el uso de diagramas UML para aspectos referentes al diseño.
Competencia 5. Capacidad de identificar, evaluar y gestionar los riesgos potenciales asociados que pudieran presentarse.	Para justificar esta competencia tendremos un control de los riesgos a lo largo de todo el ciclo de vida del proyecto. Además, en uno de los capítulos de la memoria recogeremos un "Plan de Gestión de Riesgos" tal y como hemos visto en la asignatura de "Gestión de Proyectos Software". En este artefacto se pondrán de manifiesto todos y cada uno de los riesgos que hemos visto al comenzar el proyecto y todas las amenazas que han ido surgiendo. A esto le sumamos las medidas que hemos tomado para evitarlos y así implementar una buena solución al problema a resolver.
Competencia 6. Capacidad para diseñar soluciones apropiadas en	Al utilizar una metodología ágil como SCRUM combinándolo con prototipado, estableceremos de una

uno o más dominios de aplicación utilizando métodos de la ingeniería del software que integren aspectos éticos, sociales, legales y económicos.	forma sistemática la solución del problema a través de técnicas de elicitación de requisitos, para su posterior análisis y diseño a través de diagramas UML. Además, tenemos la generación de todos los artefactos pertinentes y plasmados en la memoria del TFG, recogiendo cuáles son los stakeholders involucrados en el proyecto con la identificación de su rol, así como las tareas de las que se están encargando a lo largo de todo el proyecto. Así demostraremos la integración de aspectos sociales, legales y económicos, estableciendo cuáles son sus beneficios y responsabilidades dentro del desarrollo del producto final.

3. Objetivos.

Fruto de la motivación para la elaboración de este proyecto surgen los siguientes objetivos, aunque hemos de destacar en concreto el principal: "Implementar un sistema experto que nos permita predecir los eventos relacionados con el desgaste de un aerogenerador en base a los datos obtenidos por sus sistemas de monitorización".

Para considerar como satisfecho este objetivo hemos de alcanzar las siguientes metas a, que serían:

a) Estudio de energía eólica y familiarizarnos con el entorno de los aerogeneradores, tanto desde un punto de vista técnico, como desde uno más general.
b) Controlar el "Enterprise Service Bus" de Mule, para poder analizar el flujo de datos.
c) Familiarizarnos con el lenguaje para implementar los patrones en Esper EPL.
d) Detectar qué eventos son de interés en las medidas dadas en un aerogenerador para implementar los patrones adecuados.
e) Desarrollo de una aplicación gráfica que nos permita visualizar todos los eventos simples, y complejos fruto de los patrones implementados.
f) Generar alertas en base a los patrones implementados.

4. Método y fases de desarrollo.

En un primer lugar el desarrollo del proyecto desde el mes de enero cuenta con 23 semanas de desarrollo. En cuanto a la metodología de trabajo hemos de destacar que se usará una metodología ágil, en concreto será SCRUM, combinándolo con prototipado.

Se basará en una metodología SCRUM donde los roles están bastante bien identificados, sin

embargo, el Team, solo tiene un integrante. Por lo tanto, no podremos decir que es un SCRUM "puro" como tal. Los integrantes del grupo ajustando los roles dentro del equipo son:

- **Jefe de proyecto.** Gregorio Diaz Descalzo. Profesor dentro de la Politécnica de Albacete, en la Escuela de Ingenieros Informáticos. Coordinador entre Team y Product Owner para que haya buena comunicación y no haya ningún problema a la hora de establecer los requisitos y compromisos para y con el proyecto tanto a nivel de organización como a nivel técnico.
- **Product Owner.** Francisco José Polo Sánchez. Integrante de la empresa Ingeteam. Cliente en cuestión dentro del desarrollo del proyecto, solicita el sistema y proveerá de infraestructura de trabajo al Team. Una vez hecha la recogida y elicitación de requisitos, será el encargado en aprobarlos. En caso de haber algún error, se hallará un acuerdo y se tomarán las medidas pertinentes para su aprobación.
- **Team.** Enrique Brazález Segovia. Estudiante de 4º de Ingeniería Informática, responsable de defender el trabajo de fin de grado.

Tal y como hemos comentado anteriormente, la metodología va a estar entrelazada con el prototipado, de forma que tras acabar cada iteración en el proyecto tendremos un producto operativo y funcionando. La organización del trabajo tal y como queda esquematizado en el diagrama de Gantt de la Ilustración 2 es la siguiente:

- **Inicio del proyecto**. En esta iteración se planteará el proyecto y se interiorizarán todos los aspectos técnicos acerca del ESB, la energía eólica, etc. Además, plantearemos la planificación del proyecto, realizaremos una primera versión de los requisitos a alcanzar, y se elaborará un primer prototipo del flujo de datos usando un intermediario vía Web como es "Thingspeak". Esta aplicación externa servirá de intermediario entre el generador de eventos y el motor CEP.
- **Adaptación al caso real.** A través del ESB generaremos otro prototipo que no dependa de" Thingspeak" (a priori) y empezaremos a implementar de forma real todos y cada uno de los patrones ajustándonos al modelo de aerogenerador concreto.
- **Desarrollo de la plataforma.** Pasaremos a la elaboración de la aplicación gráfica que servirá como pantalla de visualización de los eventos detectados.
- **Tiempo de margen para la memoria y flecos sueltos.** Es una pequeña iteración que servirá como tiempo de reserva por si surge algún imprevisto que retrase el proyecto.
- **Presentación y defensa del trabajo de fin de grado.**

Ilustración 2. Diagrama de Gantt de planificación de tiempo.

5. Medios que se pretenden utilizar.

En este apartado hablaremos de todos los medios tanto software como hardware tenemos pensado utilizar para llevar a cabo el proyecto en primera instancia.

5.1. Medios Hardware.

Los medios hardware que utilizaremos para llevar a cabo el proyecto serán los siguientes:

- **Ordenador personal.** Se trata de un ordenador ASUS bastante común, utilizado durante el transcurso de la carrera. Procesador i7-5610QM CPU de 2,30 GHz y memoria RAM de 6 GB, con un tipo de sistema de 64 bits.
- **PLC de aerogenerador.** Se trata del PLC que controla todas las mediciones de las centralitas PT100 que nutren el sistema de monitorización del aerogenerador. Esta será la fuente de datos del sistema, aunque en primer lugar trataremos un conjunto de datos sintético en formato Excell o .db con el que simularemos el flujo de información, debido a que no tenemos acceso directo a ese PLC.

5.2. Medios Software.

En esta sección veremos cuáles son los elementos software que tomarán participación en nuestro entorno de trabajo. Desde los programas, IDEs y demás, hasta las librerías particulares que hacen singular este proyecto.

- **Eclipse.** Eclipse es un entorno de desarrollo integrado de Java (IDE), con gran cantidad de plugins de Open Source con lenguajes integrados como C / C++ y PHP. Puede combinar fácilmente el soporte de idiomas y otras funciones en cualquiera de los paquetes que la nube de Eclipse facilita en los paquetes predeterminados, y Eclipse Marketplace permite una personalización y extensión prácticamente ilimitadas. Hemos elegido Eclipse y no otro entorno porque el ESB está implementado sobre Eclipse, de esta forma podemos unificar

paralelamente todo el código, tanto el del simulador como el del Enterprise Service Bus [9].

- **EGit**. EGit es un proveedor de Eclipse Team para el sistema de control de versiones de Git. Git es un SCM distribuido, lo que significa que cada desarrollador tiene una copia completa de todo el historial de cada revisión del código, lo que hace que las consultas contra el historial sean muy rápidas y versátiles. El proyecto EGit está implementando herramientas Eclipse además de la implementación JGit Java de Git [10].
- **Team Foundation Service.** Se trata de un producto Software que nos permitirá llevar a cabo la gestión del trabajo. En concreto la enumeración de historias de usuario, de tareas, y generación de artefactos como sería el Sprint Burndown.
- **Anypoint Studio, ESB de Mulesoft**. Es el entorno de desarrollo de integración basado en Eclipse de MuleSoft para diseñar y probar aplicaciones Mule. Puede implementar la aplicación y ejecutarla en su servidor Mule. El mismo editor también le permite editar archivos de definición de API (en RAML y WSDL) y crear dominios que definan recursos compartidos. Anypoint Studio ofrece dos pestañas paralelas que puede utilizar para diseñar y crear sus aplicaciones: Editor visual; y un editor XML [11]**.**
- **Librería poi-3.11 para el tratamiento de documentos Excel.** Se trata de una librería de Apache para la manipulación de documentos Excel que en un principio se generarían por el PLC del aerogenerador y nutrirían al motor CEP.
- **Librería de Esper EPL.** Esper ofrece un lenguaje específico de dominio (DSL) para el procesamiento de eventos. El lenguaje de procesamiento de eventos (EPL) es un lenguaje declarativo para tratar datos de eventos basados en el tiempo de alta frecuencia. EPL cumple con el estándar SQL-92 y se amplía para analizar series de eventos y con respecto al tiempo [12].
- **Medit4CEP.** Es una herramienta para la generación de código Esper EPL, a través de un editor gráfico. Esta aplicación sirve es una solución para acercar la tecnología de procesamiento de eventos complejos a cualquier usuario independientemente del nivel de programación que posea, para que, de esta forma, se facilite la toma de decisiones [13].

Referencias.

En esta sección veremos todas las referencias bibliográficas que servirán como fuentes primarias para desarrollar en un primer momento. No obstante, puede que se vean incrementadas conforme avance el proyecto.

[1] X. Zhao, H. Fan, H. Zhu, Z. Fu, and H. Fu, "The Design of the Internet of Things Solution for Food Supply Chain," *Proc. 2015 Int. Conf. Educ. Manag. Inf. Med.*, vol. 49, no. Emim, pp. 314–318, 2015.

[2] M. Antonio and R. Borja Díaz, "Energía eólica," vol. 1, pp. 54–57, 2013.

[3] IBM, "Business Rules Management System | IBM Digital Business Automation." [Online]. Available: https://www.ibm.com/cloud/automation-software/business-decision. [Accessed: 29-Jan-2018].

[4] P. Russom, "Big data analytics," *TDWI Best Pract. Rep.*, pp. 1–35, 2011.

[5] M. Döhring *et al.*, "The convergence of workflows, business rules and complex events: Defining a reference architecture and approaching realization challenges," *ICEIS 2010 - Proc. 12th Int. Conf. Enterp. Inf. Syst.*, vol. 3 ISAS, no. January 2010, pp. 338–343, 2010.

[6] Tesis doctoral, J. Boubeta, Universidad de Cádiz. "Model-Driven Development of Domain-Specific Interfaces for Complex Event Processing in Service-Oriented Architectures.," p. 223, 2014.

[7] R. Pressman, *Ingeniería Del Software, un enfoque práctico*. 2012.

[8] AENOR, *ISO-IEC 25010*. Ginebra, 2011.

[9] Eclipse, "Eclipse - The Eclipse Foundation open source community website." [Online]. Available: https://www.eclipse.org/. [Accessed: 29-Jan-2018].

[10] Eclipse Egit, "EGit." [Online]. Available: http://www.eclipse.org/egit/. [Accessed: 29-Jan-2018].

[11] MuleSoft, "Anypoint Studio // Documentación de MuleSoft." [Online]. Available: https://docs.mulesoft.com/anypoint-studio/v/6/. [Accessed: 29-Jan-2018].

[12] EsperTech, "Esper - EsperTech." [Online]. Available: http://www.espertech.com/esper/. [Accessed: 29-Jan-2018].

[13] J. Boubeta-Puig, G. Ortiz, and I. Medina-Bulo, "MEdit4CEP: A model-driven solution for real-time decision making in SOA 2.0," *Knowledge-Based Syst.*, vol. 89, pp. 97–112, Nov. 2015.