**Implementación de solución para la gestión de licenciamiento basada en microservicios**

José Alejandro Benítez Aragón

|  |  |  |
| --- | --- | --- |
| **Información de contacto**  José Alejandro Benítez Aragón  Facultad de Ingeniería  Universidad de San Buenaventura  Especialización en Procesos de Desarrollo de Software  Cohorte XIX  Email: alejoxpi@gmail.com  Elaborado: 14/11/2020 |  | ***RESUMEN***  La comunidad OpenSource requiere para sus soluciones informáticas un sistema de gestión de la configuración orientando a la gestión de licencias de software y el seguimiento de los ítems de configuración de las licencias otorgadas a cada cliente. Esta necesidad nos lleva a la implementación de una solución de software basada en microservicios que permita de forma ágil y automatizada, la gestión de la información relacionada con la activación, validación y cambio de estados de las licencias administradas, así como sus ítems de configuración asociados, para cualquier tipo de software, sistema o modulo, cumpliendo con los atributos de calidad de alto desempeño, alta disponibilidad, facilidad de despliegue y modificabilidad. El tipo de licencia otorgado a la solución para su uso por la comunidad es MIT License.  **Palabras clave**: Licencia, Licenciamiento, Gestión, Microservicios, Configuración, Opensource |

**Implementation of a solution for microservices-based licensing management**

**ABSTRACT**

The OpenSource community requires a configuration management system for its IT solutions, guiding the management of software licenses and the monitoring of the configuration items of the licenses granted to each client. This need leads us to the implementation of a software solution based on microservices that allows, in an agile and automated way, the management of information related to the activation, validation and change of states of managed licenses, as well as their associated configuration items. for any type of software, system or module, complying with the quality attributes of high performance, high availability, ease of deployment and modifiability. The type of license granted to the solution for use by the community is MIT License.

**Keywords:** License, Licensing, Management, Microservices, Configuration, Opensource

**Introducción**

Dada la necesidad de la comunidad OpenSource de tener disponible un sistema personalizable, escalable, fácil de mantener y que pueda ser implementado en proyectos de desarrollo de diferente naturaleza, se plantea una solución orientada al desarrollo de software que permite la gestión del licenciamiento de productos, módulos y sistemas.

Las necesidades que debe resolver la solución nos encontramos con:

* Sistema orientado a la gestión de las licencias de software y su configuración.
* Permitir generar una licencia de software para un cliente o una instancia de su Sistema.
* Cada licencia debe quedar asociada a un cliente.
* Permite activar e inactivar la licencia para que las instalaciones puedan validar vía API Rest en línea si la licencia esta activa y permite trabajar.
* Recibir los ítems de configuración del cliente mediante API Rest y guardarlos.
* Mostrar los ítems de configuración recibidos para evaluar la alteración.
* Lógica de negocio expuesta como una API.
* API de servicios versionable.

Con el objetivo de aportar una solución que aborde esta necesidad, se diseña y desarrolla un sistema de diferentes componentes, de código abierto y bajo la licencia MIT, el cual puede ser reutilizado, modificado y adaptado a las necesidades particulares de cada negocio.

El código fuente de esta solución se encuentra en el siguiente enlace de GitHub:

*https://github.com/alejoxpi/license-management*

**Estrategia orientada a los microservicios**

Una “arquitectura de microservicios” es un enfoque para desarrollar una aplicación software como una serie de pequeños servicios, cada uno ejecutándose de forma autónoma y comunicándose entre sí, por ejemplo, a través de peticiones HTTP a sus API.

Normalmente hay un número mínimo de servicios que gestionan cosas comunes para los demás (como el acceso a base de datos), pero cada microservicio es pequeño y corresponde a un área de negocio de la aplicación.

Además, cada uno es independiente y su código debe poder ser desplegado sin afectar a los demás. Incluso cada uno de ellos puede escribirse en un lenguaje de programación diferente, ya que solo exponen la API (una interfaz común, a la que le da igual el lenguaje de programación en la que el microservicio esté programado por debajo) al resto de microservicios.

No hay reglas sobre qué tamaño tiene que tener cada microservicio, ni sobre cómo dividir la aplicación en microservicios, pero algunos autores como Jon Eaves caracterizan un microservicio como algo que a nivel de código podría ser reescrito en dos semanas.[[1]](#footnote-1)

**Tecnologías aplicadas**

**Azure Functions**: permite ejecutar pequeños fragmentos de código (denominados "funciones") sin preocuparse por la infraestructura de la aplicación. Con Azure Functions, la infraestructura de la nube proporciona todos los servidores actualizados que necesita para mantener la aplicación en ejecución a gran escala.

Una función se "desencadena" mediante un tipo específico de evento. Los desencadenadores admitidos incluyen responder a los cambios en los datos, responder a los mensajes, ejecutarse según una programación o como resultado de una solicitud HTTP.

Aunque siempre puede programar directamente en una gran cantidad de servicios, la integración con otros servicios se simplifica mediante el uso de enlaces. Los enlaces proporcionan acceso declarativo a una gran variedad de servicios de Azure y de terceros.[[2]](#footnote-2)

**MongoDB:** es un sistema de base de datos NoSQL, orientado a documentos y de código abierto. En lugar de guardar los datos en tablas, tal y como se hace en las bases de datos relacionales, MongoDB guarda estructuras de datos BSON (una especificación similar a JSON) con un esquema dinámico, haciendo que la integración de los datos en ciertas aplicaciones sea más fácil y rápida. MongoDB es una base de datos adecuada para su uso en producción y con múltiples funcionalidades. [[3]](#footnote-3)

**Angular**: es un framework para aplicaciones web desarrollado en TypeScript, de código abierto, mantenido por Google, que se utiliza para crear y mantener aplicaciones web de una sola página. Su objetivo es aumentar las aplicaciones basadas en navegador con capacidad de Modelo Vista Controlador (MVC), en un esfuerzo para hacer que el desarrollo y las pruebas sean más fáciles. Angular se basa en clases tipo "Componentes", cuyas propiedades son las usadas para hacer el binding de los datos. En dichas clases tenemos propiedades (variables) y métodos (funciones a llamar).

**Secuencia de acciones**

A continuación, se enumeran la secuencia de pasos principal que define a grandes rasgos el comportamiento de la solución para ejecución de la activación, consulta y validación de las licencias otorgadas.

**Secuencia lógica**

1. Se crean los códigos de activación para un producto, modulo, componente o funcionalidad. Los códigos de activación definen el tipo y duración de la licencia.
2. Se entrega el código de activación al cliente.
3. Se activa la licencia con el código de activación.
4. El sistema o software a licenciar debe implementar el consumo de la API para activar la licencia enviando parámetros de configuración.
5. El sistema o software que haga uso del sistema de validación de licencias debe consumir la API enviando el código de licencia y los parámetros usados para su activación. Este proceso comprueba si la licencia esta activa y es válida.
6. El sistema o software cliente debe interpretar la respuesta para desactivar su uso.

**Resultado**

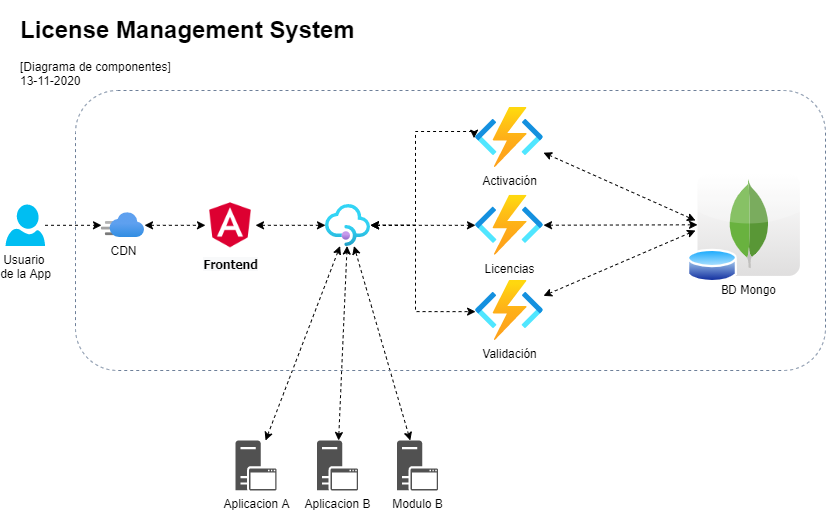


Figura 1. Diagrama de componentes

El resultado obtenido como solución a la necesidad de la comunidad OpenSource es la implementación de una aplicación de software basada en la arquitectura microservicios que puede ser desplegada en contenedores tipo docker o alojada en la nube de Azure. A través de las Azure Functions se expone la funcionalidad de la lógica de negocio a través de endpoints API Rest que permiten a nivel de integración con otros sistemas y del frontend, hacer uso de la lógica implementada.

|  |  |
| --- | --- |
| Contrato de datos activación licencia | Contrato de datos validación licencia |
| {      "activationcode":"33d4-553-4b6b-a060-7a43e",      "activationsettings":      {          "hardwareid":"13545-45454-56456",          "email":"test@gmail.com",          "company":"Universidad San Buenaventura",          "location":"Cali-Colombia",          "customercode":"26f4-4215-4894-ade4-1958f7"      }  } | {      "licensecode":"a5d2c56c9fbcf633f88102ab4cea889f70",      "validationsettings":      {          "hardwareid":"13545-45454-56456",          "email":"test@gmail.com",          "company":"Universidad San Buenaventura",          "location":"Cali-Colombia",          "customercode":"26f4-4215-4894-ade4-1958f7"      }  } |

Tabla 1. Contrato de datos enviados en las peticiones a la REST API

Para lograr una visualización más cómoda de la información, se despliega un frontend desarrollado en Angular que consume las diferentes APIs expuestas por los microservicios.

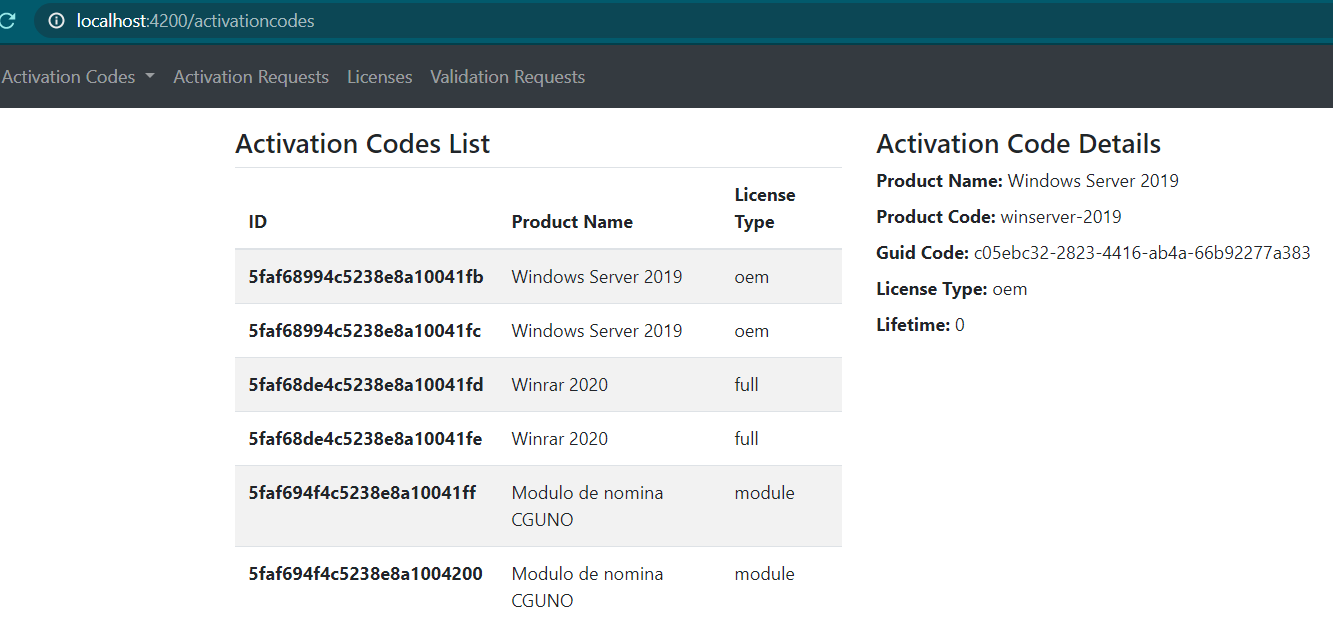


Figura 2. Pantalla de lista de códigos de activación

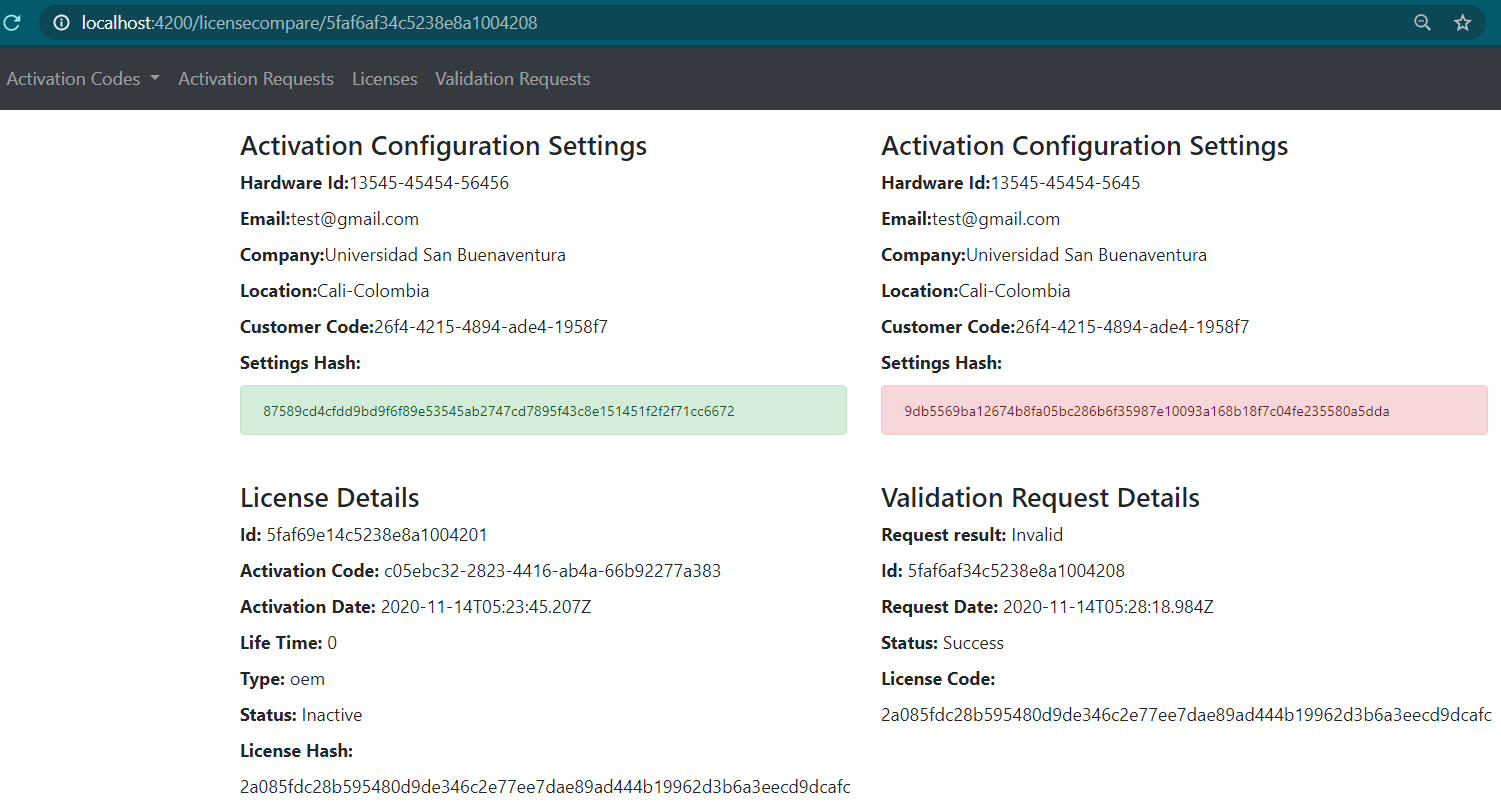


Figura 3. Pantalla de comparación de una licencia vs una solicitud de validación fallida

Finalizando, el soporte de persistencia o almacenamiento de datos lo suministra una base de datos aprovisionada sobre el motor MongoDB.

**Conclusión**

El desarrollo de software OpenSource es clave para el éxito del ecosistema tecnológico es claro y conocido. Empresas de todos los sectores de actividad adoptan esta metodología de desarrollo por los beneficios que aporta no solo relativas a las conocidas libertades en las que se basa el software libre, sino por los valores de seguridad, colaboración y de transparencia que ofrece a la hora de trabajar tanto desde el punto de vista de los clientes, como desarrolladores.

Aportar soluciones que faciliten a los desarrolladores integrar sistemas sencillos, escalables y personalizables para sus necesidades de gestión del licenciamiento, permiten de cierto modo retribuir todo ese conocimiento y beneficios que la comunidad ofrece.

**Referencias**

¿Qué es eso de los microservicios?, 2020, [ https://bit.ly/3lvKaMh ]  
Introducción a Azure Functions, 2020, [https://docs.microsoft.com/es-mx/azure/azure-functions/functions-overview]  
MongoDB, 2020, [https://es.wikipedia.org/wiki/MongoDB]

1. ¿Qué es eso de los microservicios? [ <https://bit.ly/3lvKaMh> ] [↑](#footnote-ref-1)
2. Introducción a Azure Functions [https://docs.microsoft.com/es-mx/azure/azure-functions/functions-overview] [↑](#footnote-ref-2)
3. MongoDB [https://es.wikipedia.org/wiki/MongoDB] [↑](#footnote-ref-3)