Universidad San Jorge

Escuela de Arquitectura y Tecnología

Grado en Ingeniería Informática

Proyecto Final

Seguridad K8s

Autor del proyecto: Andrea Álvaro Martín

Director del proyecto: María Francisca Pérez

Tutor en empresa: NTT DATA

Zaragoza, 25 de junio de 2023

Este trabajo constituye parte de mi candidatura para la obtención del título de Graduado en Ingeniería Informática por la Universidad San Jorge y no ha sido entregado previamente (o simultáneamente) para la obtención de cualquier otro título.

Este documento es el resultado de mi propio trabajo, excepto donde de otra manera esté indicado y referido.

Doy mi consentimiento para que se archive este trabajo en la biblioteca universitaria de Universidad San Jorge, donde se puede facilitar su consulta.

Firma Fecha **7 julio 2023**

Dedicatoria y Agradecimiento

FALTA

1. Introducción 3 paginas

Tabla de contenido

Tabla de ilustraciones

Tabla de tablas

Resumen

En los últimos años, la adopción de arquitecturas basadas en microservicios ha aumentado significativamente, lo que ha creado una necesidad de herramientas de seguridad especializadas para proteger plataformas basadas en Kubernetes (K8s). Por lo tanto, el objetivo de este proyecto es analizar dos herramientas de seguridad para K8s: Falco y StackRox. Falco es una herramienta de prevención de intrusiones que se integra directamente con Kubernetes y se enfoca en detectar comportamientos inusuales en tiempo real. Por otro lado, StackRox es una plataforma de seguridad para Kubernetes que se enfoca en la administración de políticas de seguridad y cumplimiento, detección de amenazas y respuesta automatizada.
La seguridad en Kubernetes es crucial ya que una brecha de seguridad puede tener graves consecuencias como pérdida de datos o interrupción del servicio, lo que dañaría la reputación de la organización. Los entornos de Kubernetes son altamente dinámicos y escalables, lo que los hace vulnerables a las amenazas de seguridad. Es necesario contar con herramientas actualizadas y efectivas para detectar y mitigar los riesgos de seguridad de manera oportuna en la nube.
El proyecto se divide en dos partes. En la primera sección, se realizará un estudio técnico de las herramientas de seguridad disponibles para Kubernetes, con un enfoque en Falco y StackRox para determinar cuál es el mejor en cada caso de uso. En la segunda sección, las herramientas elegidas para la demostración se desempaquetarán en un entorno Kubernetes, y se creará una guía de instalación para los usuarios.

Palabras clave: *Kubernetes, StackRox, Falco, Seguridad, microservicios*

Abstract

In recent years, the adoption of microservice-based architectures has increased significantly, creating a need for specialized security tools to protect Kubernetes (K8s) based platforms. Therefore, the objective of this project is to analyze two security tools for K8s: Falco and StackRox. Falco is an intrusion prevention tool that integrates directly with Kubernetes and focuses on detecting unusual behaviors in real time. On the other hand, StackRox is a security platform for Kubernetes that focuses on security policy management and compliance, threat detection and automated response. Security at Kubernetes is crucial as a security breach can have serious consequences such as data loss or service interruption, which would damage the organization's reputation.

Kubernetes environments are highly dynamic and scalable, making them vulnerable to security threats. Updated and effective tools are needed to detect and mitigate security risks in the cloud in a timely manner.

The project is divided into two parts. In the first section, a technical study of the safety tools available for Kubernetes will be made, with a focus on Falco and StackRox to determine which is the best in each use case. In the second section, the tools chosen for the demonstration will be unpacked in a Kubernetes environment, and an installation guide will be created for users.

Keywords: Kubernetes, StackRox, Falco, Security, microservices

2. Introducción 3 paginas

"¿Cuál es el problema?"
Debe centrar al lector en la problemática del Proyecto desarrollado.

Debería empezar contextualizando la importancia del proyecto dentro de una problemática general que se describa brevemente.

La introducción debería terminar indicando la organización del proyecto para acometer el proyecto a desarrollar.

En la actualidad, la tecnología de contenedores se ha convertido en una herramienta fundamental para el despliegue de aplicaciones y servicios en la nube. En este contexto, Kubernetes se ha consolidado como la plataforma de orquestación de contenedores más popular y ampliamente utilizada. Sin embargo, la seguridad en Kubernetes es un tema crítico que ha recibido mucha atención en los últimos años.

El problema radica en que Kubernetes, al igual que cualquier otra tecnología, tiene vulnerabilidades y debilidades que pueden ser explotadas por atacantes malintencionados. La naturaleza distribuida y escalable de Kubernetes también lo hace más complejo de proteger y monitorear adecuadamente. Además, dado que Kubernetes es utilizado en entornos empresariales críticos, cualquier violación de la seguridad puede tener consecuencias graves.

Es por ello que resulta crucial desarrollar estrategias y herramientas de seguridad para proteger adecuadamente las implementaciones de Kubernetes. De esta manera, se pueden minimizar los riesgos de ataques, fugas de información o interrupciones de servicio, asegurando así la integridad y disponibilidad de los sistemas.

Para abordar esta problemática, el proyecto que se desarrollará tiene como objetivo investigar y comprar de forma exhaustiva las herramientas existentes desde el nivel de clúster hasta el nivel de aplicación, tanto en términos de plataforma como de código.

El proyecto se lleva a cabo en colaboración con la empresa NTTDATA para mejorar la seguridad de los datos en la nube en los entornos de Kubernetes.

2. Antecedentes / Estado del Arte

Hoy en día son más numerosas las organizaciones y empresas que buscan integrar un diseño a gran escala y de alto rendimiento. Es por ello por lo que muchas de ellas se decantan por una arquitectura enfocada a microservicios.

2.1. De sistemas monolíticos a microservicios

En los últimos años, los sistemas monolíticos han sido la principal opción utilizada por las empresas, y muchas siguen utilizándolos. Estos sistemas se fundamentan en una base de funcionalidades centralizada, donde todas las funcionalidades y servicios de negocio están agrupados en una base de código única. Entre sus principales ventajas destacan su facilidad de desarrollo inicial, desarrollo centralizado que puede establecer un estándar, y su simplicidad a nivel de despliegue y ejecución.

Sin embargo, para realizar un cambio en este tipo de arquitectura, es necesario actualizar todo el código base, creando y desplegando una versión actualizada de la aplicación. Esto hace que las actualizaciones sean restrictivas y consuman más tiempo de desarrollo, pruebas y despliegue. Además, se dificulta la adopción de nuevas tecnologías, se requiere un despliegue completo tras cada actualización y los fallos pueden propagarse por todo el sistema.

Aunque la arquitectura monolítica todavía es común para muchas aplicaciones, no es la mejor opción para sistemas complejos. A medida que el software evoluciona y crece, el costo de mantener una arquitectura única se vuelve cada vez mayor, mientras que los beneficios de adoptar una arquitectura más flexible para respaldar e impulsar el crecimiento empresarial son aún mayores. La arquitectura de microservicios surgió como una respuesta de los desarrolladores a la incapacidad de continuar escalando aplicaciones.

Por ello, las arquitecturas de microservicios resuelven estos problemas mencionados, permitiendo así una sencillez en el desarrollo, test y despliegue, gran escalabilidad horizontal, aislamiento de errores, total adaptabilidad a nuevas tecnologías.

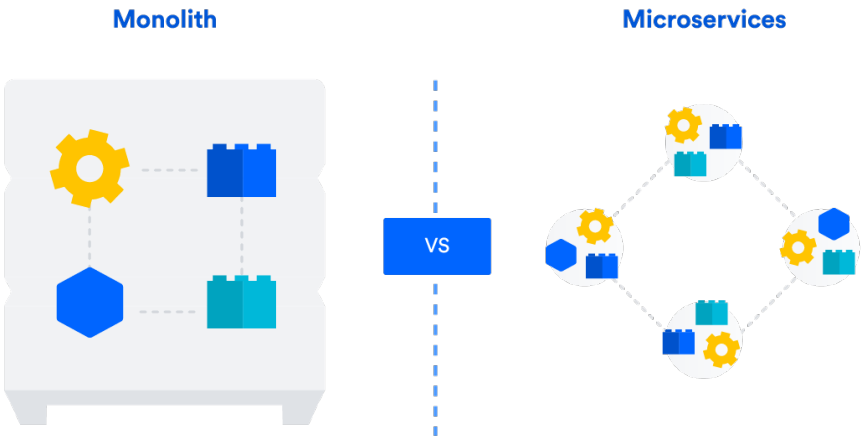

Figura 1. *Comparativa de Sistema Monolíticos y Microsistemas*

Las arquitecturas de microservicios resuelven los problemas mencionados, permitiendo un sencillo desarrollo, pruebas y despliegue, una gran escalabilidad horizontal, aislamiento de errores y total adaptabilidad a nuevas tecnologías.

No obstante, los microservicios también plantean desafíos que deben ser abordados. Uno de ellos es la necesidad de gestionar múltiples servicios, lo que puede aumentar la complejidad para los desarrolladores, operadores y DevOps. Además, es necesario coordinar los servicios entre sí para garantizar su correcto funcionamiento.

El diseño de aplicaciones basadas en microservicios puede ser complicado y requiere un diseño adecuado de los microservicios individuales. También puede haber complejidad en la consistencia de los datos y transacciones, ya que cada servicio tiene su propia base de datos. A medida que aumenta la necesidad de automatización, también aumenta la necesidad de una supervisión rigurosa.

Otro aspecto importante a considerar es el aumento de los riesgos de seguridad, ya que cada función se expone al exterior a través de una API, lo que resulta en un mayor número de posibles vectores de ataque. Esto ha llevado a la búsqueda de soluciones y análisis de amenazas en este ámbito.

Empresas destacadas como Netflix, Spotify o Google utilizan arquitecturas basadas en microservicios en muchos de sus servicios y aprovechan Docker para acelerar el desarrollo y la implementación de aplicaciones. Esta arquitectura se utiliza para implementar y ejecutar entornos en la nube, que ofrecen numerosas ventajas y oportunidades, pero también conllevan riesgos.

A medida que aumenta la complejidad de la infraestructura utilizada en la nube, también aumentan los riesgos, lo que significa que los ciberdelincuentes están aprovechando estas oportunidades para llevar a cabo ataques. Por lo tanto, es fundamental abordar la seguridad en Kubernetes, la plataforma de orquestación de contenedores de código abierto más utilizada para administrar aplicaciones basadas en microservicios.

2.2. Contenerización y Orquestación de contenedores

La tecnología de despliegue de aplicaciones basada en contenedores ha sido uno de los mayores avances en la industria del software en los últimos años. Debido a la necesidad de acelerar la actualización continua de aplicaciones y la irrupción de la metodología DevOps, estos despliegues se han vuelto cada vez más comunes. Muchas organizaciones y empresas se han inclinado por el uso de contenedores debido a su facilidad para el despliegue de nuevos servicios, la escalabilidad y la capacidad de adaptarse a cualquier entorno.

Los contenedores son una forma de virtualización del sistema operativo que permite utilizar un único contenedor para ejecutar todo, desde microservicios o procesos de software hasta grandes aplicaciones. Contienen todos los archivos ejecutables, códigos binarios, bibliotecas y archivos de configuración necesarios, pero a diferencia de los métodos de virtualización de computadoras o servidores, no contienen imágenes del sistema operativo, lo que los hace más ligeros y reduce considerablemente el costo. En despliegues de aplicaciones más grandes, se pueden implementar varios contenedores como uno o más clústeres de contenedores.

Los contenedores ofrecen un mecanismo de empaquetado lógico que permite abstraer las aplicaciones del entorno en el que se ejecutan, lo que facilita y uniformiza el despliegue de aplicaciones basadas en contenedores. Si se compara con las máquinas virtuales, los contenedores permiten empaquetar las aplicaciones con bibliotecas y otros programas, proporcionando entornos aislados para ejecutar servicios software y simplificando aún más el entorno.

En resumen, los contenedores ofrecen una solución mucho más ligera que permite a los desarrolladores y equipos de operaciones de IT trabajar con mayor facilidad y disfrutar de ciertos beneficios.

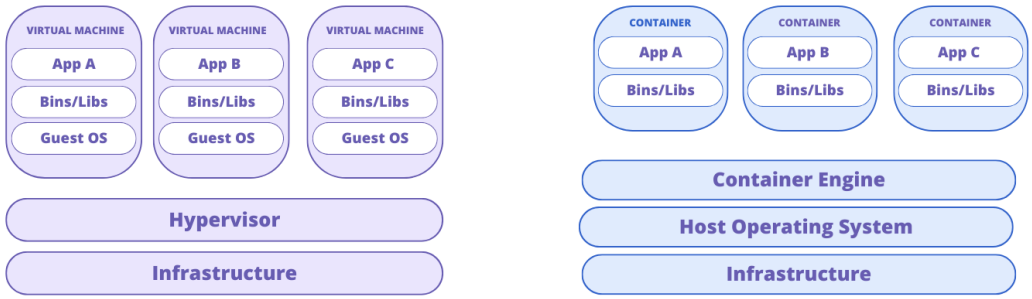

Figura 2. *Comparativa de Virtualización y Contenirización*

La integración de los contenedores y orquestadores de contenedores, así como la monitorización del sistema, permite que el despliegue de los servicios en un entorno de producción se haga de manera fiable y robusta.

Los contenedores aportan las características y herramientas concretas que se necesitan en la actualidad, donde la portabilidad, escalabilidad, alta disponibilidad y los microservicios en aplicaciones distribuidas son cada vez más utilizados. Cada vez se desarrollan menos aplicaciones monolíticas y más basadas en módulos y microservicios. Esto permite un desarrollo más ágil, más rápido y simultáneamente portátil.

2.2.1. Herramientas existentes

Docker

Docker es la plataforma de open source más utilizada para la contenerización, y permite a los usuarios empaquetar, distribuir y administrar aplicaciones dentro de contenedores. Con Docker, podemos implementar y escalar rápidamente sus aplicaciones a cualquier entorno con la confianza de que su código se ejecutará en cualquier lugar. Docker es una plataforma de software que permite la contenerización de aplicaciones y servicios, lo que facilita el desarrollo, la implementación y la gestión de aplicaciones en cualquier entorno. Con Docker, los desarrolladores y equipos de operaciones de IT pueden implementar aplicaciones y servicios de forma rápida y eficiente, sin tener que preocuparse por las complejidades de la infraestructura subyacente.

Kubernetes

Kubernetes es una plataforma de orquestación de contenedores de código abierto que permite la gestión automatizada de aplicaciones en contenedores en un entorno de nube o local. Fue desarrollado por Google y se convirtió en un proyecto de la Cloud Native Computing Foundation. Kubernetes proporciona un conjunto de características para desplegar, escalar y gestionar aplicaciones contenerizadas de manera eficiente y segura.

Algunas de sus características principales incluyen el escalado automático de aplicaciones, la gestión de la configuración, la tolerancia a fallos, el balanceo de carga y la gestión de recursos. También admite múltiples proveedores de nube y sistemas operativos, lo que lo hace altamente portátil y adaptable a diferentes entornos.

Según un informe de *Cloud Native Computing Foundation* publicado en 2020, Kubernetes se ha convertido en la plataforma de orquestación de contenedores dominante en la nube, con el 83 % de las empresas encuestadas. Además, el informe encuentra que el 78 % de las organizaciones que utilizan Kubernetes lo utilizan en un entorno de producción.

Otro estudio de *Datadog* publicado en 2021 encuentra que el uso de Kubernetes ha aumentado significativamente en los últimos años. Según su informe, el uso de Kubernetes aumentó un 37 % en 2020 en comparación con 2019 y su adopción sigue creciendo.

Además, según una encuesta de Red Hat de 2021, el 96 % de los encuestados cree que Kubernetes es importante para el éxito a largo plazo de su negocio y el 89 % planea expandir su uso de Kubernetes en el futuro.

Por lo tanto, estos datos indican que Kubernetes se adopta y usa ampliamente en todo el mundo y que su adopción y uso continúan creciendo en la actualidad.

2.2.2. DevOps y Soluciones de seguridad en el mercado actual

Las empresas tienen la responsabilidad de mantener el sistema operativo actualizado e instalar los parches de seguridad que sean necesarios en todo momento. Al igual que ocurre en los servidores que son propiedad de una empresa, es necesario mantener políticas de seguridad tradicionales como el control de usuarios, la correcta configuración de servicios, o la revisión del software para comprobar que no tiene vulnerabilidades, entre otros aspectos.
Para intentar solventar los problemas de seguridad en los microservicios, han surgido diversas herramientas para automatizar y gestionar la seguridad de dichos entornos de una manera más rápida y eficaz.
Las vulnerabilidades de software nos han acompañado desde los inicios de la informática, y en los últimos tiempos la preocupación por las mismas ha ido en aumento en todos los ámbitos debido a los daños materiales, económicos e incluso a la reputación que pueden ocasionar. Los ciberdelincuentes y profesionales de la ciberseguridad estudian las vulnerabilidades con el fin de explotarlas en el caso de los ciberdelincuentes, y darles una solución, mitigar los daños o prevenir una posible incidencia en el caso de los profesionales de la ciberseguridad.
En un entorno DevOps basado en contenedores, las organizaciones deben abordar tres problemas de seguridad clave. En primer lugar, detectar vulnerabilidades en las aplicaciones tanto en el código fuente de la aplicación como en dependencias externas; por ejemplo, componentes open source. En segundo lugar, flexibilidad, es decir, que no requieran perímetros externos o configuración de redes. Finalmente, deben garantizar la seguridad durante todo el ciclo de vida del desarrollo de software.

Para este cometido han surgido los perfiles DevSecOps, que se centran en añadir seguridad al ciclo de vida de software tanto en el despliegue como monitorización de los sistemas.

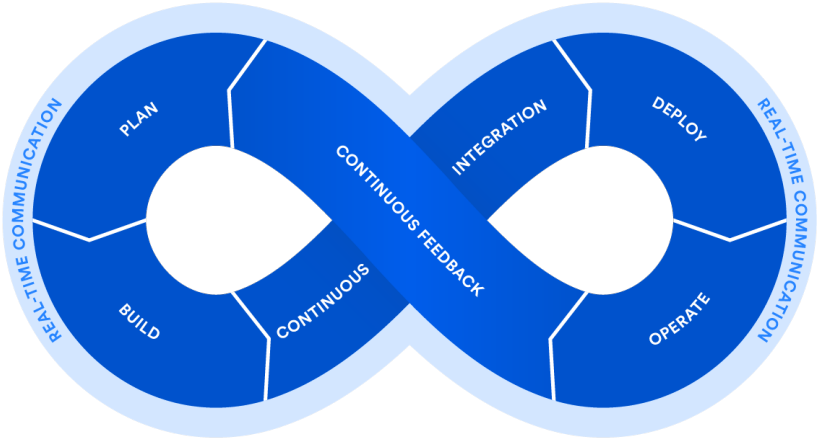

Figura 3. *Ciclo de desarrollo del Software en DevOps*

La seguridad de los contenedores es importante porque la imagen del contenedor contiene todos los componentes que eventualmente ejecutarán la aplicación. Si una imagen de contenedor contiene vulnerabilidades, aumenta el riesgo y la gravedad potencial de los problemas de seguridad durante la producción y puede ser catastrófico para una empresa u organización.

Actualmente en el mercado existen diversas herramientas y soluciones que proporcionan a las empresas llevar a cabo un ciclo de vida de software seguro.
Esta es una clasificación de herramientas de seguridad existentes por categorías.

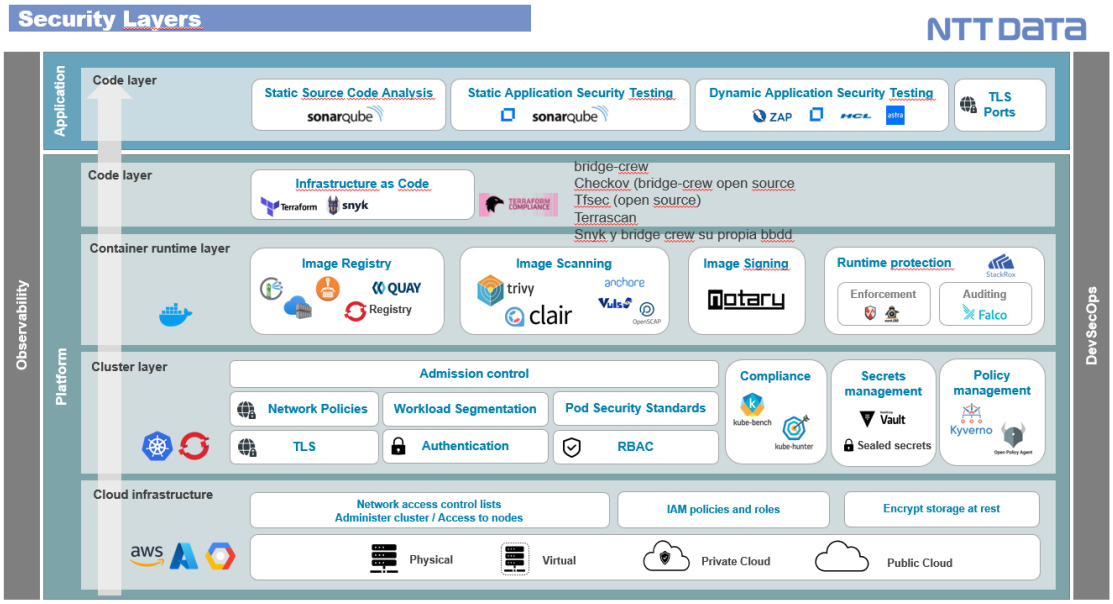

Figura 4. *Clasificación de herramientas de seguridad existentes*

En este proyecto mencionaremos la importancia de los niveles de seguridad y cuáles son las más empleadas además de una breve comparativa. Y nos focalizaremos en el apartado de *Runtime protection* analizando en profundidad las herramientas como Falco y StackRox.

2.2.3. Principales herramientas analizadas

Falco

Falco es una solución de seguridad nativa de la nube basada en Kubernetes que analiza las llamadas al sistema y la actividad de las aplicaciones en tiempo real para detectar y prevenir problemas de seguridad. Emplea un conjunto de reglas para especificar el comportamiento anticipado y advierte sobre las violaciones, con acciones como detener los procesos ofensivos que son personalizables. Falco se basa en la tecnología Sysdig y puede identificar una amplia gama de riesgos en los sistemas Kubernetes, mejorando la seguridad y la defensa contra futuros ataques.

StackRox

StackRox es una tecnología de seguridad de contenedores que ofrece seguridad continua para la configuración de Kubernetes. A través de capacidades como la gestión de vulnerabilidades, la gestión de cumplimiento y el perfil de riesgos, identifica y mitiga las amenazas de seguridad. StackRox detecta y responde a los riesgos de seguridad utilizando el aprendizaje automático y se conecta con las tecnologías DevOps para una implementación y administración sin problemas.

3. Objetivos

Los objetivos principales que se definieron para este proyecto, que están recogidos en la propuesta de este [Anexo 1], son:

- Configurar y desplegar las herramientas de seguridad Sysdig Secure, Falco y StackRox.
- Realizar una comparativa de las herramientas en base a las funcionalidades que ofrecen.
- Crear documentación que permita reproducir lo realizado y configurar las principales buenas prácticas de seguridad sobre las herramientas.

Durante la primera fase del proyecto se analizaron estos objetivos y se establecido el alcance del proyecto.

Tras el análisis de los antecedentes, analizamos las herramientas mencionadas en los objetivos. Sin embargo, concluimos que uno de los principales problemas era una de las herramientas propuestas, Sysdig Secure, puesto que era de pago.

Por ello, se reajustan los objetivos del proyecto con la empresa NTTDATA. Y se propone un estudio general de las herramientas existentes para segurizar Kubernetes en diferentes niveles. Además de hacer una breve conclusión.

El primer objetivo de este proyecto es una vista general de las herramientas existentes para segurizar Kubernetes y el despliegue y configuración de las herramientas Falco y StackRox.

El segundo objetivo de este proyecto consiste en realizar un estudio comparativo entre las herramientas Falco y StackRox.

Por último, este proyecto tiene como objetivo la creación de una documentación que permita reproducir lo realizado y configurar las principales buenas prácticas de seguridad de ambas herramientas,

4. Metodología

Este proyecto se va a desarrollar como Trabajo Fin de Grado, asignatura estimada en 12 créditos ECTS lo que supone, aproximadamente, 320 horas de trabajo a desarrollar por, en este caso, una sola persona. La fecha de entrega de este proyecto es el 25 de junio de 2023, por lo que la flexibilidad para la iteración es limitada.
El proyecto se combina con otras asignaturas y trabajos, y el tiempo disponible para su desarrollo no es completo. La disponibilidad para trabajar en este proyecto cambiará de una semana a otra en función de la carga de trabajo externa que se tenga en el momento. Sin embargo, estimo un mínimo de 20 horas semanales. Por tanto es necesaria una metodología flexible y de respuesta rápida frente a imprevisto.
Teniendo en cuenta las características descritas, se elige Extreme Programming (XP) como metodología para este proyecto.

4.1 Extreme Programming (XP)

Extreme Programming es una metodología de desarrollo de software ágil que enfatiza en el trabajo en equipo, la comunicación, la simplicidad, la retroalimentación y la mejora continua. Está diseñada para entregar software de alta calidad de manera oportuna y rentable, proporcionando un marco para gestionar y responder al cambio durante el proceso de desarrollo. La metodología implica el desarrollo iterativo, la integración continua, pruebas frecuentes y el uso de estándares y buenas prácticas de programación para asegurar la entrega de un software de alta calidad.

Se eligió la metodología ágil XP para el desarrollo de este proyecto debido a su enfoque en la refactorización del código y la retroalimentación, así como en el fomento de buenas prácticas y el uso de diversos elementos considerados importantes. La promoción de la refactorización y la retroalimentación durante todo el proceso de desarrollo es especialmente beneficioso para el Objetivo 2 del proyecto, que implica un estudio comparativo de dos herramientas, ya que hasta que no se hayan analizado de manera técnica y práctica, no se conocerán las decisiones más acertadas.

La metodología XP se elige también por su enfoque en el uso de buenas prácticas, ya que este es el primer proyecto profesional del equipo y se considera una buena oportunidad para adoptar hábitos positivos. Entre los elementos más útiles de XP para este proyecto se encuentran las historias de usuario, las pruebas de aceptación, las estimaciones, la planificación de iteraciones y el registro de comunicaciones entre el cliente y los desarrolladores.

XP se considera especialmente adecuada para proyectos con requisitos imprecisos y altamente cambiantes, así como para aquellos con un alto riesgo técnico, como puede ser el caso de la falta de experiencia en el desarrollo de software para este proyecto.

4.2 Aplicación de la tecnología

XP, al igual que la mayoría de las técnicas ágiles, se crea para un pequeño equipo de desarrolladores; en consecuencia, la primera diferencia que se observa en el empleo de este enfoque es que este equipo de desarrollo consta de una sola persona. Esto implica la eliminación de reuniones o *stand ups* diarias.

En XP, la figura del cliente es altamente significativa; en este caso, NTTDATA ocupará esta posición, y su evaluación e ideas guiarán el progreso del proyecto.

Otro cambio significativo es que las iteraciones ya no serán definidas por un número de días. Como se indicó anteriormente, el tiempo disponible para este proyecto es variable. Como resultado, al comienzo de cada iteración, se hará una estimación del tiempo disponible en función de la carga de trabajo externa, y se determinará la duración de la siguiente iteración.

4.3 Seguimiento del desarrollo

Se utilizarán otras herramientas para ayudar a la gestión de proyectos, además de los aspectos empleados por XP para el desarrollo de software.

4.3.1 Trello

Para realizar el seguimiento se ha empleado la herramienta Trello. He seleccionado esta herramienta puesto que estoy familiarizada con ella ya que la he empleado en diversos proyectos durante los estudios académicos y a nivel empresarial.

Trello consiste en una herramienta visual compuesta por un tablero con varias columnas en las que puedes crear, mover eliminar tarjetas, de esta forma conseguimos de forma visual conocer el estado actual de nuestro proyecto además de conseguir mayor eficacia en cuanto a tiempo empleado en la herramienta.

En nuestro caso, al tratarse de un proyecto grande y al trabajar por iteraciones, trabajaremos de la siguiente forma. Al inicio de cada iteración las historias del usuario se dividirán en tareas más pequeñas, cada tarea será una tarjeta y se añadirá una estimación y una prioridad. A medida que se vayan completando las tareas y desarrollando la iteración las tarjetas se irán cambiando de columnas o estado hasta ser completadas.

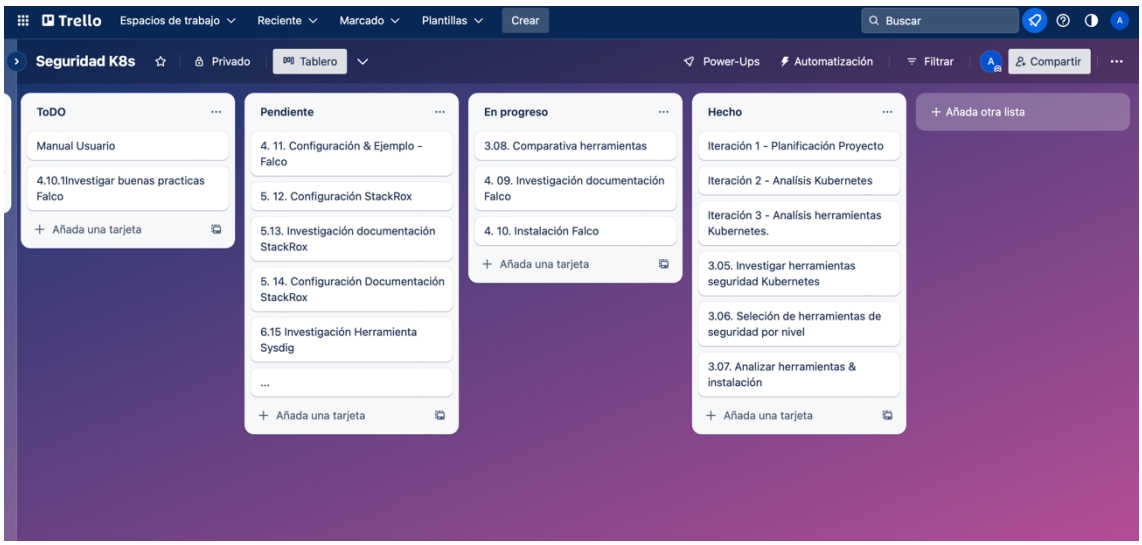

Figura 5. Trello´s Dashboard del proyecto

4.3.2 Diagrama de trabajo

Para lograr una visualización global más clara del proyecto, se han dividido los tiempos en iteraciones y se ha elaborado un diagrama de Gantt utilizando una plantilla de Excel. El diagrama registra el tiempo estimado para cada iteración, permitiendo un análisis detallado de la duración real de cada una. Además, se lleva un registro más minucioso de cada iteración, con el objetivo de comparar los tiempos estimados con los tiempos reales y mejorar las estimaciones futuras a lo largo del proyecto.

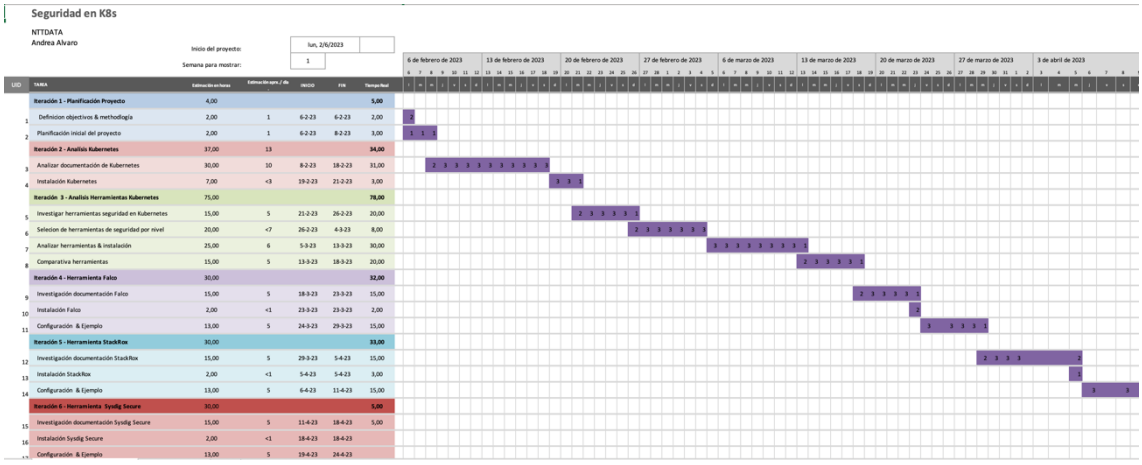

Figura 6. Diagrama de Gantt del proyecto

Como es de observar ha habido desviaciones de acuerdo a la estimación inicial del proyecto.

Figura . Diagrama de Gantt del trabajo real. – FINAL PROYECTO

3. Implementación / Estudio / Desarrollo

Análisis

Análisis del problema

Diseño

Implementación

4. Estudio económico

"¿Cuál es el presupuesto necesario para la realización de este proyecto?"

"¿Qué beneficios económicos va a crear en la empresa (real o ficticia)?"

4.1. Desglose de costes

4.1.1. Costes materiales

4.1.2. Costes humanos

4.1.3. Costes de infraestructura

4.1.4. Costes totales

5. Resultados

"¿Qué datos/productos se han obtenido al finalizar el proyecto?"

Resultados de simulaciones y procesos del Proyecto. Se deben reflejar las pruebas y test realizados, errores detectados...

Artefactos producidos.

Desviaciones de la metodología, acciones correctivas, riesgos eliminados o mitigados a lo largo de la vida del proyecto.

6. Conclusiones

"¿Lo realizado se corresponde con lo previsto?"

En este apartado se debe reflejar el grado de cumplimiento del objetivo o de los objetivos planteados.

Se deben indicar , si existen, las posibles mejoras y futuras ampliaciones del objetivo del Proyecto.

6.1. Propuesta de mejora

7. Bibliografía

[1] «GHC Peñalara - Software generador de horarios para centros educativos,» Peñalara Software S.L., [En línea]. Available: https://www.penalara.com/es/ES. [Último acceso: Junio 2020].

[1] [En linea] Available: https://kubernetes.io/docs/concepts/services-networking/network-policies/
[2] [En linea] Available: https://docs.tigera.io/calico/3.25/getting-started/kubernetes/self-managed-onprem/onpremises

https://www.atlassian.com/es/microservices/cloud-computing/containers-vs-vms

8. Anexos