**FORMATO PARA EL DESARROLLO DE COMPONENTE FORMATIVO**

| PROGRAMA DE FORMACIÓN | Aseguramiento de la calidad de *software* |
| --- | --- |

| COMPETENCIA |  | RESULTADOS DE APRENDIZAJE |  |
| --- | --- | --- | --- |

| NÚMERO DEL COMPONENTE FORMATIVO | 19 |
| --- | --- |
| NOMBRE DEL COMPONENTE FORMATIVO | Automatización de pruebas |
| BREVE DESCRIPCIÓN | En este componente formativo se resalta la importancia que tiene la automatización de pruebas para detectar fallos en el *software.* En este proceso, el *software* es sometido a pruebas 24/7, en los que se seleccionan algunos procesos para ser monitoreados y evitar pruebas manuales, por medio de herramientas de automatización. |
| PALABRAS CLAVE | Virtualización, entornos virtuales, plan de pruebas, cargas estáticas, cargas dinámicas. |

| ÁREA OCUPACIONAL | 2 - CIENCIAS NATURALES, APLICADAS Y RELACIONADAS |
| --- | --- |
| IDIOMA | Idioma principal utilizado en el contenido del recurso |

1. **TABLA DE CONTENIDOS:**

**Introducción**

1. **Entorno de pruebas**
   1. Máquinas virtuales
   2. Contenedores
   3. Dimensionamiento de pruebas de escalabilidad
2. **Ejecución de *scripts* de pruebas**
3. **INTRODUCCIÓN**

Cordial bienvenida a este componente formativo, en el que se tratará la automatización de pruebas*.* Para conocer la importancia del tema, lo invitamos a conocer el siguiente video:

| CF19\_Introduccion |
| --- |

1. **DESARROLLO DE CONTENIDOS:**
2. **Entorno de pruebas**

El término automatización, es un concepto que se refiere a la ejecución de tareas repetitivas de manera más ágil y eficientemente, con lo cual es posible ahorrar tiempo y dinero; se busca automatizar las pruebas para identificar los fallos de aplicación y corregir a tiempo alguna irregularidad en los sistemas. Para tomar la decisión de automatizar es necesario conocer tanto el contexto de la empresa como las herramientas y las técnicas apropiadas, para que este proceso arroje los resultados esperados. En este sentido, se revisarán los entornos de pruebas.

Según Choucair (2022), los ambientes de prueba y datos bien administrados son la base para implementar soluciones de negocio; son claves dentro de las operaciones de tecnología e información (TI) y facilitan la entrega de productos al mercado, de manera ágil y con calidad.



Estos entornos utilizan dispositivos *hardware* y *software* para probar los productos en desarrollo, por lo general se cuenta con una réplica exacta de la aplicación que se quiere testear, con la cual es posible generar los resultados deseados y compararlos con los resultados esperados. Así, los entornos de prueba juegan un rol importante en la mitigación de grandes costos en una compañía y se pueden configurar según la prueba seleccionada.

Existen dos formas de clasificar un entorno de prueba que están marcando la pauta a nivel mundial:

| CF19\_1\_Formas |
| --- |

Los métodos de prueba que se utilizan en el ambiente de pruebas, dependerán de los objetivos que se quieren alcanzar. Por ejemplo, el equipo de desarrollo ágil de *software* se apoya en varias cajas de arena en el ambiente de prueba, incluyendo la caja de arena para el aseguramiento de la calidad, una para pruebas y otra para demostraciones.

Otras propuestas de prueba son las siguientes:

* Desarrollo de prototipos.
* Scrum.
* Modelo en espiral.
* Modelo en cascada.

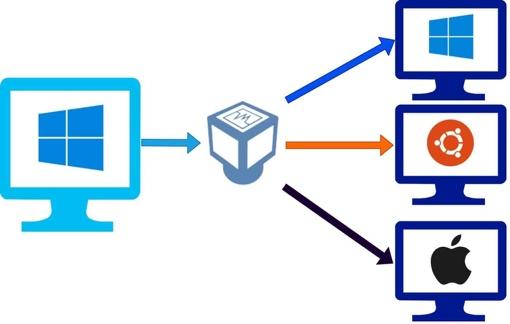
Además, la relevancia de la implementación de un ambiente retarda según los requerimientos de los clientes, cuando las empresas son grandes y su infraestructura tecnológica es robusta, la exigencia de implementar un buen entorno de prueba es mucho más alta, por lo que las personas involucradas en el proyecto se deben acoplar y sintonizar para el desarrollo armónico de las pruebas, que se pueden configurar mediante una gestión del entorno de prueba (TEM).

A continuación, se presentan los diferentes componentes que hacen parte del entorno de pruebas.

1. **Máquinas virtuales**

Una máquina virtual es un sistema operativo virtual con su propia memoria, interfaz de red y almacenamiento; estas máquinas se crean en un sistema de *hardware* físico y un hipervisor.

Pero, ¿qué se entiende por hipervisor y cómo funciona este sistema?



* ***Host:*** las máquinas físicas que se equipan con el hipervisor, se denominan *host.*
* **Hipervisor:** es el *software* utilizado para la separación de los recursos *hardware*, que posteriormente utilizará la máquina virtual.
* ***Guest:*** se denomina *guest* a las diversas máquinas virtuales que utilizan los recursos.

Así, los ***hosts***utilizan los recursos *hardware* del computador para redistribuirlos entre los ***i*** existentes o las nuevas máquinas virtuales.

Una de las características de las máquinas virtuales es la ejecución de varios sistemas operativos en un mismo *hardware*, por ejemplo, alguna distribución de Linux en un portátil con sistema operativo MacOS, de manera que la experiencia final del usuario es como si estuviera trabajando directamente en un computador físico con su propio sistema operativo.

La virtualización tiene la propiedad de compartir un sistema completo con varios entornos virtuales, de allí que el hipervisor juegue un papel muy importante, debido a que es el encargado de administrar los entornos físicos de las máquinas virtuales, según la demanda desde este entorno físico, hasta la máquina virtual; tiene la capacidad de agendar las solicitudes de las máquinas virtuales y de acceder a más recursos físicos solicitados y asignarlos según la urgencia.

Estás máquinas pueden estar aisladas del resto del sistema, pero también pueden coexistir en una misma computadora como un servidor, por lo cual se mencionan dos tipos de hipervisores:

| CF19\_1\_1\_Hiperversores |
| --- |

De acuerdo con lo que se ha expuesto, es posible preguntar: **¿por qué utilizar las máquinas virtuales?**



Actualmente, estas máquinas son muy populares en las empresas, porque influyen sustancialmente en el factor económico; en este sentido, evitan invertir en más costos para compra de *hardware,* teniendo en cuenta que es posible aprovechar las bondades de las MV y utilizar una misma máquina, en la que se incorporen varios servidores, utilizando los recursos *hardware* para optimizar el rendimiento. Asimismo, su implementación permite la conmutación de la información en catástrofes, sin incurrir en gastos adicionales.

Se debe tener en cuenta lo siguiente:

Las máquinas virtuales generan un entorno aislado de manera que lo que ocurre en este entorno no infiere con el resto de la máquina o host. En los entornos de desarrollo son muy utilizadas para probar aplicaciones nuevas sin que estas afecten el correcto funcionamiento de otras MV y en sí del hardware donde se encuentran alojadas.

Es muy importante que se considere el **dimensionamiento** cuando se operan las máquinas virtuales, debido a que este es clave para definir el tamaño de nuestra virtualización y el alcance que se le va a dar para que el proyecto tenga la mejor respuesta a los usuarios. Por ello, se debe tener presente que si se realiza un cálculo de dimensionamiento sin datos, los resultados pueden ser inexactos; en este sentido se aconseja hacer este proceso con una carga de datos máxima, con una periodicidad de una hora y, lo más importante, generar una caracterización o perfil de una transacción. Por este motivo, se insiste en determinar el volumen máximo de transacciones por hora o, en el caso de existir varios momentos, se necesitará detallarlos y especificarlos para obtener un dimensionamiento adecuado.

De acuerdo con lo que se ha tratado, es necesario tener presente que cuando se establece la carga que se realiza a una infraestructura, es importante entender algunos factores que intervienen en ello:

| CF19\_1\_1\_Factores |
| --- |

1. **Contenedores**

El tema de contenedores se relaciona directamente con el de la virtualización de un sistema operativo; así es que en un solo contendedor se puede ejecutar cualquier aplicación, microservicio, aplicativo o algo más robusto.

Un contenedor contiene los ejecutables, el código binario, bibliotecas y los archivos de configuración, de manera que hay una seria comparación con el propósito de las máquinas virtuales; los contenedores no trabajan con el sistema operativo o por lo menos principalmente, esta característica los hace más rápidos y preferidos, con una utilización del sistema mucho menor. Cuando se ponen a prueba aplicaciones robustas se pueden ejecutar varios contenedores como si fuese un *clúster* de contenedores, los equipos se pueden administrar mediante un orquestador de contenedores como *kubernetes*.

Las ventajas que presentan los contenedores frente a las máquinas virtuales, son las siguientes:

| CF19\_1\_2\_Ventajas |
| --- |

Los contenedores hoy en día son muy conocidos gracias a la aparición de una herramienta de fácil manejo y muy versátil para los desarrolladores alrededor del mundo llamada ***Docker***, que ha logrado integrar muchos desarrolladores geográficamente distantes y trabajar juntos en un mismo proyecto, utilizando las bondades de los contenedores, para realizar la creación, prueba y puesta en marcha optimizadas, a partir de las mismas imágenes de contenedores.

Entonces, ¿qué es un *docker*?



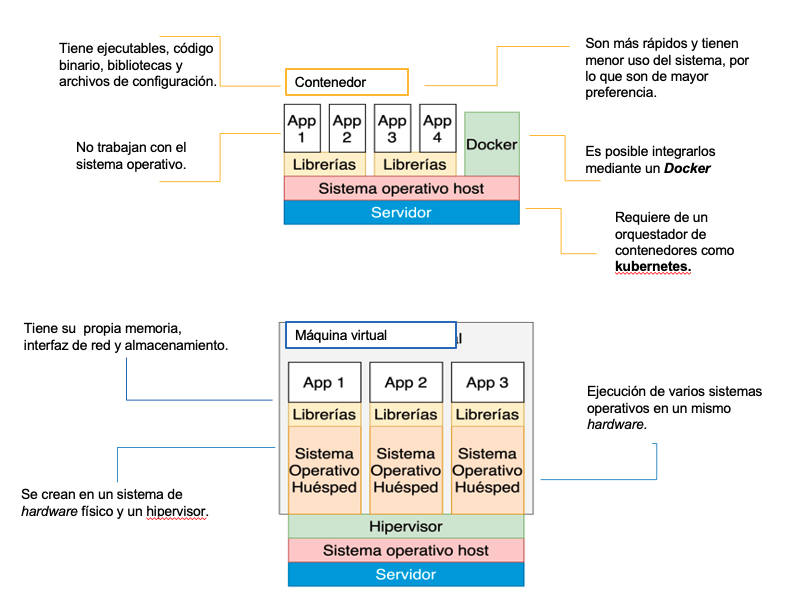


Dentro de los entornos de desarrollo, en el mundo de los contenedores, es común encontrar aplicaciones robustas que quizás tengan cientos o miles de contenedores; para que la aplicación pueda cumplir con su objetivo es necesario emplear los famosos orquestadores llamados **kubernetes,** que es una herramienta diseñada para la ejecución en cadena de varios contenedores de algún *software,* incluido el ***Docker***.

Después de conocer en qué consiste cada uno de estos componentes, se deben revisar las diferencias más significativas entre las máquinas virtuales y los contenedores:

**Figura 1**

*Comparación entre las máquinas virtuales y los contenedores*



* 1. **Dimensionamiento de pruebas de escalabilidad**

Para probar la escalabilidad, se deben considerar las métricas más importantes tales como el tiempo de respuesta, latencia, rendimiento y utilización de recursos, de manera que se pueda estimar la capacidad de ampliación de la arquitectura, sin perder la calidad de la prestación de los servicios.

Las métricas de evaluación de escalabilidad seleccionadas son:

| CF19\_1\_3\_Metricas |
| --- |

Una vez se definen las métricas para la estimación de la escalabilidad de una arquitectura, se determinan los tipos de pruebas a establecer, según las recomendaciones Microsoft y Oracle, pero omitiendo las pruebas de funcionalidad conforme con el interés especial del modelado, el diseño de procesos de negocio y la simplicidad de estos procesos diseñados:

| CF19\_1\_3\_Procesos |
| --- |

De esta forma, quedan determinadas las métricas y las pruebas de evaluación de escalabilidad en la arquitectura solución.

**Metodología para una evaluación de escalabilidad**

Según Pinchao (2011), la metodología descrita tanto por Oracle (2010) como por Microsoft (2007) es muy similar en sus procedimientos; sin embargo, por limitantes de recursos tecnológicos y tiempo de la metodología a emplear, no se hace una comparación entre cada una de las iteraciones de las pruebas (como proponen Oracle y Microsoft), dejando el análisis para el conjunto completo de las pruebas.

Las fases que componen esta metodología son las que se presentan a continuación:

| CF19\_1\_3\_Fases |
| --- |

Se pueden aplicar diferentes tipos de pruebas como las de escalabilidad y las de carga:

* Las **pruebas de escalabilidad** analizan la forma en la que responde la aplicación ante la concurrencia simultánea de clientes; se busca que las aplicaciones se ajusten a los requerimientos de los usuarios simultáneos y que utilicen los recursos de manera lógica para que la experiencia del usuario sea la mejor frente a la usabilidad del sistema. Así, estas pruebas se pueden implementar en *hardware,* recursos de red y bases de datos, para analizar su comportamiento en un número de solicitudes simultáneas.
* En las **pruebas de carga** se somete al sistema a diferentes tipos de carga; cuando se realizan las pruebas de carga al sistema, se evalúa la escalabilidad del mismo a varios niveles de carga que se le proporcionan, esta última prueba es importante porque es común utilizarla en los ya conocidos entornos en contenedores.

El dimensionamiento de las pruebas de escalabilidad se puede llevar a cabo a nivel de *hardware,* *software* o base de datos; los datos o parámetros que se utilizan para la medición difieren de una solicitud a otra, por ejemplo, para una página web, podría ser la cantidad de usuarios, CPU o la red; en cambio a un servidor web se le mediría el número de solicitudes procesadas.

Los atributos que se manejan en una prueba de escalabilidad para determinar su dimensionamiento, son los siguientes:

| CF19\_1\_3\_Atributos |
| --- |

Al realizar esta prueba, se comprenderá el comportamiento del cliente cuando se le asigna la carga máxima y qué formas existen para resolverlo, también permite conocer el desgaste del servidor y los tiempos de respuesta frente a la cantidad de usuarios de la aplicación.

Ahora, es necesario identificar algunos aspectos importantes que se deben tener en cuenta para la creación de escenarios, para probar la escalabilidad de una aplicación:

| CF19\_1\_3\_Aspectos |
| --- |

1. **Ejecución de *scripts* de pruebas**

Un *script* de prueba es un conjunto de instrucciones en un sistema para realizar determinadas pruebas utilizando unos parámetros para analizar resultados y que estos sean los esperados.

Con el fin de mostrar la manera en la que se realiza esta ejecución de *scripts* de prueba, se realizará el siguiente ejercicio, en el cual partimos de las características que se presentan en tres computadores, con el fin de revisar la forma de realizar la ejecución de *scripts* de pruebas.

Para cada uno de los tres (3) equipos se observó el Sistema Operativo SO, la memoria RAM, la alta definición HD y la unidad central de procesamiento CPU.

**Tabla 1**

*Identificación de los equipos*

| **Equipo** | **SO** | **RAM** | **HD** | **CPU** |
| --- | --- | --- | --- | --- |
| Equipo1 | Windows 8.1 | 4 GB | 500 GB Mecánico | Core I3 2.5 GHz |
| Equipo2 | Windows 10 | 8 GB | 480 GB SSD | Core I5 2.6 GHz |
| Equipo3 | Windows 11 | 12 GB | 1 Tera | Core I7 3.0 GHz |

Para comenzar, se elaboran las pruebas de carga, en las que se hace la primera evaluación sobre los servicios con un *host*, desplegando los servicios de ESB y un elemento de carga (otros *hosts*) con los parámetros de carga para tal fin.

Antes de presentar los valores de carga se describen los parámetros de configuración de los algoritmos de carga, así:

* **Hilos:** cada hilo que se configura es un proceso de ejecución que hace parte del proceso de negocio.
* **Periodo ente hilos:** define cada cuántos segundos disparará cada uno de los hilos, es decir, la concurrencia.
* **Número de veces de ejecución:** determina las veces que se ejecutarán determinados números de hilos o la recurrencia.

En la siguiente tabla, observamos los parámetros de carga y la configuración de clúster y de carga para la prueba: se presenta el ID de prueba para cada equipo y se relaciona con el *host* de carga, el número de hilos por grupo, el periodo entre grupos de hilos por segundo y el número de ejecuciones.

**Tabla 2**

*Parámetros de carga, configuración de clúster y de carga para la prueba*

|  | **Equipo** | ***Host* de carga** | **Número de hilos por grupo** | **Periodo entre grupo de hilos (seg)** | **Número de ejecuciones** |
| --- | --- | --- | --- | --- | --- |
| **P1.1** | Equipo1 | 1 | 120 | 30 | 3 |
| **P1.2** | 1 | 180 | 20 | 6 |
| **P1.3** | 1 | 240 | 10 | 9 |
| **P2.1** | Equipo2 | 2 | 240 | 20 | 3 |
| **P2.2** | 2 | 380 | 10 | 6 |
| **P2.3** | 2 | 500 | 5 | 9 |
| **P3.1** | Equipo3 | 2 | 600 | 20 | 3 |
| **P3.2** | 2 | 900 | 10 | 6 |
| **P3.3** | 2 | 1000 | 5 | 9 |

Como hemos ya indicado, con estas pruebas evaluamos las pruebas de carga. Pero también es importante establecer una comparación entre ambientes de Arquitectura Orientada a Servicios, SOA y no SOA.

Veamos, entonces, los resultados de las pruebas de carga para cada equipo. Analicemos los datos de latencia, *throughout*, los porcentajes en el disco duro y en memoria RAM.

**Tabla 3**

*Resultados de la prueba de carga para cada equipo*

| **Equipos** | **Id de prueba** | **Latencia**  **(ms)** | ***Throughout***  **(tps)** | **Disco duro %** | | | | **Memoria ram (%)** | | | |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Equipo1** | P1.1 | 1933 | 273 | 36.3 | | | | 86.6 % | | | |
| P1.2 | 2223 | 452 | 36.7 | | | | 88.3 % | | | |
| P1.3 | 2536 | 467 | 39.9 | | | | 91.1 % | | | |
| **Equipo2** | P2.1 | 1702 | 587 | 46.8 % | | 38.3 % | | 91.2 % | | 82.2 % | |
| P2.2 | 1493 | 522 | 52.2 % | | 38.7 % | | 92.8 % | | 74.6 % | |
| P2.3 | 1587 | 656 | 46.2 % | | 39.9 % | | 84.2 % | | 65.9 % | |
| **Equipo3** | P3.1 | 1202 | 684 | 66.2 % | 45.6 % | | 35.2 % | 92.2 % | 72.7 % | | 82.5 % |
| P3.2 | 1194 | 721 | 62.1 % | 58.2 % | | 41.6 % | 96.5 % | 81.3 % | | 61.2 % |
| P3.3 | 1025 | 735 | 62.1 % | 56.2 % | | 44.3 % | 93.3 % | 63.8 % | | 62.1 % |

Ahora, se procede a realizar las pruebas de estrés, para lo cual se emplean los siguientes ítems sobre la arquitectura, con el fin de determinar los límites del sistema, manteniendo constante el periodo entre hilos con un valor de 5 segundos.

Número de hilos por grupo: 120, 180, 240…..

Número de ejecuciones: 3, 6, 9, 12…….

En estos resultados, observamos la prueba P.5.n, del equipo 1, en donde más del 20 % de la ejecución del proceso produce error de denegación de servicio. De esta misma forma se procederá con las pruebas cuando en el equipo existan 2 nodos, para lo cual el ID de la prueba será P.6.x y pruebas P.7x para 3 nodos.

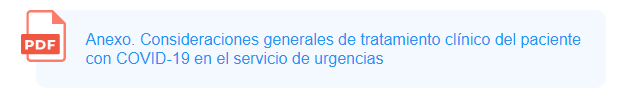
**Tabla 4**

*Resultados de determinadas pruebas*

| **Id de prueba** | **Equipo** | ***Host* de carga** | **Número de hilos por grupo** | **Periodo entre grupo de hilos (seg)** | **Número de veces de ejecución** |
| --- | --- | --- | --- | --- | --- |
| **P51.1** | 1 | 1 | 120 | 5 | 3 |
| **P51.2** | 1 | 180 | 5 | 6 |
| **P51.3** | 1 | 240 | 5 | 9 |

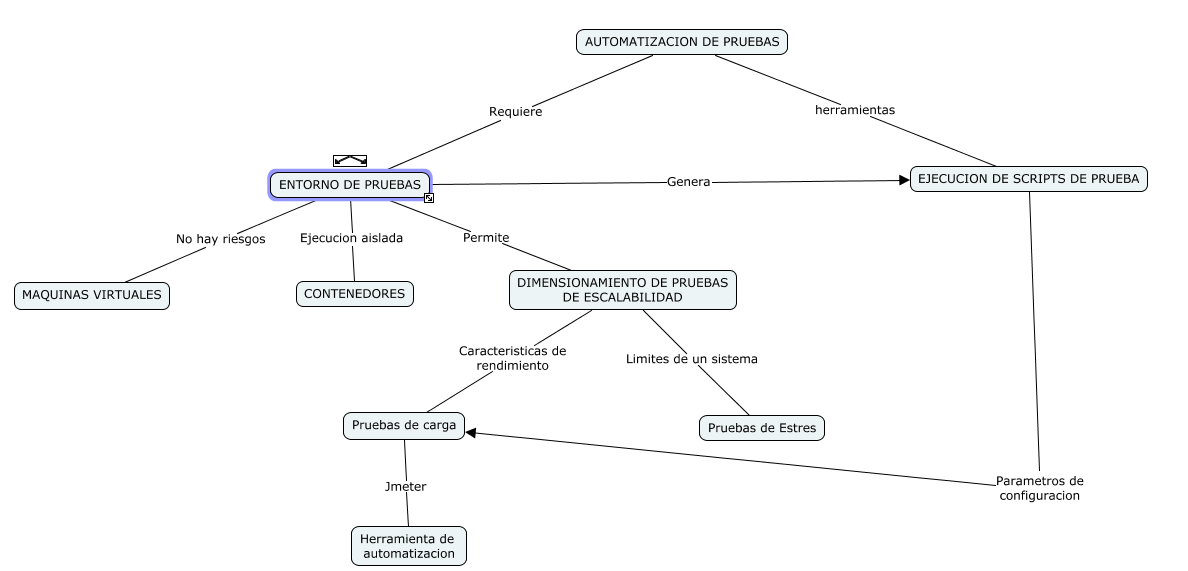
El trabajo de diseñar y ejecutar pruebas de flexibilidad, es una labor ardua que requiere de personal experto para determinar en el servidor de aplicaciones, diversas configuraciones de los nodos de los equipos (número de sesiones concurrentes, número de conexiones del pool de conexiones, memoria de los objetos en el servidor, entre otros), para determinar las causas de la denegación de servicio. También se requiere la colaboración de administradores de bases de datos para sistemas de almacenamiento.

Con el fin de familiarizarse con la forma en la que se obtienen los resultados de las pruebas, se le invita a revisar y a analizar el siguiente documento:



1. **SÍNTESIS**

A continuación, se presenta una síntesis de la temática estudiada en el componente formativo:



1. **ACTIVIDADES DIDÁCTICAS (Se debe incorporar mínimo 1, máximo 2)**

| DESCRIPCIÓN DE ACTIVIDAD DIDÁCTICA | |
| --- | --- |
| Nombre de la Actividad | Relacionar palabras relacionadas con la automatización de pruebas |
| Objetivo de la actividad | Identificar algunas palabras relacionadas con las especificaciones técnicas y pruebas de calidad de *software.* |
| Tipo de actividad sugerida | Emparejamiento |
| Archivo de la actividad  (Anexo donde se describe la actividad propuesta) | CF19\_Actividad\_didactica.docx |

1. **MATERIAL COMPLEMENTARIO:**

| Tema | Referencia APA del Material | Tipo de material  (Video, capítulo de libro, artículo, otro) | Enlace del Recurso o  Archivo del documento o material |
| --- | --- | --- | --- |
| Automatización de pruebas | De los santos, J. (2020). *Todo lo que necesitas saber antes de automatizar pruebas* [video]. YouTube. | Video | <https://www.youtube.com/watch?v=dGDmhbeYfWQ> |
| Máquinas virtuales | Google Cloud LATAM. (2021). *Máquinas virtuales en la nube* [video]. YouTube. | Video | <https://www.youtube.com/watch?v=H7EAHuRbIy0> |
| Ejecución de *scripts* de pruebas | The QA Testing Channel. (2017). *¿Qué es un Script Automatizado? - en 1 minuto* [video]. YouTube. | Video | <https://www.youtube.com/watch?v=xt1NnslQ9MA> |

1. **GLOSARIO:**

| TÉRMINO | SIGNIFICADO |
| --- | --- |
| Atributos: | un atributo de calidad es una propiedad medible de un sistema, que indica qué tan bien el sistema satisface las necesidades de las partes interesadas (Zapata, 2020). |
| *Docker:* | el sistema de *software* de TI llamado *Docker* es la tecnología de organización en contenedores que posibilita la creación y el uso de los contenedores de Linux(RedHat, 2018). |
| *Guests:* | la máquina física original en que está instalada el hipervisor se llama *host*, y las VM que utilizan estos recursos se llaman *guests*. Los *guests* utilizan los recursos informáticos como la CP, la memoria y el almacenamiento, como un conjunto de medios que pueden redistribuirse fácilmente(RedHat, 2018). |
| Hipervisor: | un hipervisor, conocido también como monitor de máquinas virtuales, es un proceso que crea y ejecuta máquinas virtuales. Un hipervisor permite que un ordenador *host* preste soporte a varias máquinas virtuales invitadas mediante el uso compartido virtual de sus recursos, como la memoria y el procesamiento (Vwware, s.f.). |
| *Host:* | la máquina física original en que está instalado el hipervisor se llama *host* virtualización (Redhat, s.f.). |
| Kernel o núcleo: | es una parte fundamental del sistema operativo que se encarga de conceder el acceso al *hardware* de forma segura para todo el *software* que lo solicita. El *Kernel* es una pequeña e invisible parte del sistema operativo, pero la más importante, porque sin esta no podría funcionar. Todos los sistemas operativos tienen un *Kernel*, incluso Windows 10, pero quizá el más famoso es el *Kernel* de Linux, que ahora además está integrado en Windows 10 con sus últimas actualizaciones (Geeknetic, s.f.). |
| Kubernetes: | *Kubernetes* es una plataforma portable y extensible de código abierto para administrar cargas de trabajo y servicios. *Kubernetes* facilita la automatización y la configuración declarativa. Tiene un ecosistema grande y en rápido crecimiento. El soporte, las herramientas y los servicios para *Kubernetes* están ampliamente disponibles (Kubernetes, s.f.). |
| *Script* de prueba: | es un conjunto de instrucciones para probar automáticamente una aplicación. Los casos de prueba se utilizan para el entorno de prueba manual. La prueba se utiliza en el entorno de prueba automatizado (Ebook online, s.f.). |

1. **REFERENCIAS BIBLIOGRÁFICAS:**

Choucair. (s.f.). *¿Sabes qué son los ambientes de prueba de desarrollo de software?* <https://www.choucairtesting.com/a-day-at-the-office>

Ebooks online. (s.f.). *¿Qué es un guion de prueba? Cómo escribir con el ejemplo****.***

Erickson, E., Brady, G. y Olprod (2022). *Requisitos de dimensionamiento de hardware para entornos locales.* Microsoft. <https://docs.microsoft.com/es-es/dynamics365/fin-ops-core/fin-ops/get-started/hardware-sizing-on-premises-environments>

Kubernetes (s.f.). *¿Qué es Kubernetes****?*** <https://kubernetes.io/es/docs/concepts/overview/what-is-kubernetes>

Geeknetic. (s.f.). *¿Qué es el Kernel y para qué sirve?* <https://www.geeknetic.es/Kernel/que-es-y-para-que-sirve>

NetApp. (s.f.). *¿Qué son los contenedores?* <https://www.netapp.com/es/devops-solutions/what-are-containers>

Myservername.com. (s.f.). *Los 10 mejores sitios y programas de software gratuitos para compartir archivos en 2021*. <https://es.myservername.com/10-best-free-file-sharing-sites#1_Response_Time>

Pinchao, P. M. (2011). Propuesta para la interoperabilidad de procesos de negocios en sistemas de información hospitalarios bajo una arquitectura orientada a servicios. [Monografía para optar al título de Ingeniero en Electrónica y Telecomunicaciones, Universidad del Cauca].

RedHat (2019). *¿Qué es una máquina virtual?* <https://www.redhat.com/es/topics/virtualization/what-is-a-virtual-machine#descripci%C3%B3n-general>

# RedHat. ( 2018). *¿Qué es DOCKER?* <https://www.redhat.com/es/topics/containers/what-is-docker>

Vmware. (s.f.). *¿Qué es un hipervisor?* <https://www.vmware.com/latam/topics/glossary/content/hypervisor.html>

Zapata, M. (2019). *Atributos de calidad del software*. <https://manuelzapata.co/atributos-de-calidad>

1. **CONTROL DEL DOCUMENTO**

|  | Nombre | Cargo | Dependencia | Fecha |
| --- | --- | --- | --- | --- |
| Autor (es) | José Luis Bastidas Pérez | Experto Temático | Regional Cauca – Centro de Teleinformática y Producción Industrial | Abril 2022 |
| Ana Catalina Córdoba Sus | Evaluadora instruccional | Regional Antioquia - Centro de Servicios de Salud | Marzo 2024 |
| Olga Constanza Bermúdez Jaimes | Responsable Línea de Producción Antioquia | Regional Antioquia - Centro de Servicios de Salud | Marzo 2024 |

1. **CONTROL DE CAMBIOS**

**(Diligenciar únicamente si realiza ajustes a la Unidad Temática)**

|  | Nombre | Cargo | Dependencia | Fecha | Razón del Cambio |
| --- | --- | --- | --- | --- | --- |
| Autor (es) |  |  |  |  |  |