

Tecnología en Desarrollo de Software

Unidad 2.

Tipos de sistemas operativos



Unidad 2. Tipos de sistemas operativos

¿Sabías que los sistemas operativos han evolucionado de forma constante y se encuentran en todos nuestros dispositivos electrónicos?

Dependiendo de la finalidad y la perspectiva con la que se aborde su estudio, pueden darse múltiples clasificaciones: desde su estructura, pasando por los servicios o el tipo de aplicaciones que integran la cantidad de usuarios que utilizan simultáneamente el sistema, hasta los dispositivos hacia los que se encuentran enfocados.

Los avances tecnológicos son notorios en la actualidad, y con ellos la evolución de los sistemas operativos; de esta forma, la clasificación de los mismos se hace cada vez más vasta y requieren un estudio minucioso para orientar el desarrollo de software en ellos.

Es por eso que en esta unidad estudiaremos diferentes tipos de sistemas operativos desde el enfoque de su aplicación o el hardware al que están orientados: dispositivos móviles o sistemas embebidos.

Podremos concluir, al finalizar la unidad, que cada plataforma o sistema de hardware tiene asociada una respectiva plataforma de software.



Tema
1

Sistemas operativos para dispositivos móviles

¿Sabes de dónde surgen los dispositivos móviles?

Los dispositivos móviles surgen como una derivación de los computadores portátiles, viéndose notablemente favorecidos por los avances tecnológicos en miniaturización (Wolf, Ruiz, Bergero, & Meza Vega, 2014).

Uno de los más notorios sistemas operativos que incursiona en los asistentes digitales personales y luego en los dispositivos móviles o teléfonos móviles de la primera década del siglo XXI es Symbian. Tiene sus orígenes en el sistema operativo EPOC de Psion (Retamosa de Ágreda, 2009).



Symbian

Su desarrollo fue liderado por Nokia. En 1998 se origina una alianza alrededor del sistema operativo con participantes como Motorola, Siemens, Panasonic, Fujitsu, Samsung, LG, Sony Ericsson, Sanyo, entre otros. Este era un grupo que buscó competir con Windows Mobile al originar Symbian Ltd.

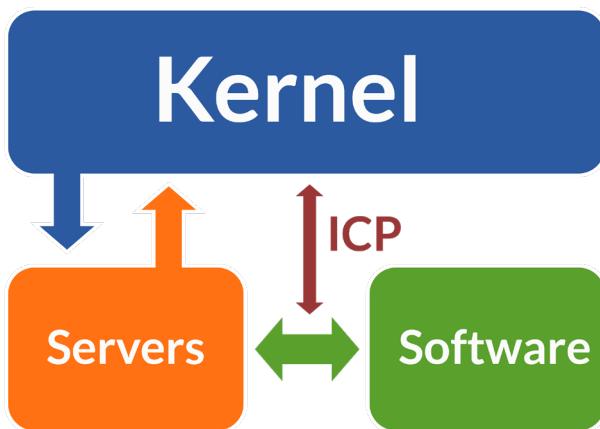
Symbian se consolidó como sistema común para múltiples dispositivos, aprovechado por diversos fabricantes de hardware, ofreciendo un uso optimizado de recursos (muy limitados en comparación con una estación de cómputo tradicional), como gestión de memoria RAM y batería.

El lenguaje de programación nativo de este sistema operativo es C++, el cual, a diferencia de lenguajes como JAVA o Visual Basic, ofrece acceso a todas las funcionalidades del entorno. Para el desarrollo de nuevas aplicaciones, se cuenta con un SDK que permite personalizar el entorno de trabajo y manipular funcionalidades como gestión de agenda, telefonía y multimedia. Es un sistema monousuario, específicamente optimizado para procesadores de la familia ARM (Advanced RISC Machines).

Cuenta con diferentes API para manejo de datos, textos, gráficos, y maneja los protocolos de comunicación TCP/IP, GSM y WAP.

Permite transferencia de datos de forma serial y por Bluetooth. Su navegador de internet soporta HTML.





Estudiemos otros datos importantes del Symbian:

- Su arquitectura está basada en microkernel y el modelo cliente-servidor.
- La comunicación inter-procesos (IPC) es un mecanismo que permite que diferentes procesos se comuniquen con el núcleo de forma independiente o separada, por medio de mensajes y manejo de hilos de procesamiento (la comunicación es concurrente, es decir, se pueden dar múltiples comunicaciones simultáneamente).
- Los servicios carecen de interfaz gráfica y se ejecutan en segundo plano a manera de *demonio* y permiten interactuar directamente con el hardware.
- Actualmente, el código fuente del sistema operativo se encuentra disponible por medio de una Licencia Pública Eclipse (EPL).

En detalle, según West & Wood (2014), la arquitectura modular de Symbian se divide en:

- **Interfaz de hardware y servicios de kernel.** Gestiona la memoria y administra la multitarea.
- **Servicios base y de sistema operativo.** Administra servicios de telefonía, comunicación serial, redes, conectividad y gráficos.
- **Servicios de aplicación.** Permite desplegar aplicaciones desarrolladas en JAVA; desde esta capa se despliega JAVA J2ME, y se tiene la posibilidad de manejar aplicaciones ofimáticas como procesadores de texto, soporte de sincronización de datos, administración de información personal (PIM).
- **Framework de interfaz de usuario.** Despliega y controla el escritorio del sistema.

El paradigma y la interacción con el usuario cambian con la incursión disruptiva de Apple en el mercado de los dispositivos móviles con su iPhone y su respectivo sistema operativo, iPhone OS.

Windows Phone y Blackberry OS son algunos de los sistemas operativos que no se sobrepusieron al nuevo paradigma impuesto por el diseño predominante de Apple.

El iPhone OS está basado en un sistema tipo Unix: Darwin, base fundamental de MAC OS. Su arquitectura actual se muestra en la siguiente imagen donde se visualiza un modelo en capas (Uribe 2010).

- Cocoa Touch

Capa de abstracción que permite a un usuario interactuar con el sistema; incluye reconocimiento de gestos multitáctiles, visualización de notificaciones y gestión de componentes visuales; su principal lenguaje de programación es Objective-C, y cuenta con un framework que permite desarrollar aplicaciones nativas.

- Media

Abarca los servicios que permiten desplegar componentes multimedia (audio, video, gráficos).

- Core Services

Comprende los servicios que utiliza el sistema, fundamentalmente recuperación y escritura de información en base de datos local (Sqlite); gestiona las comunicaciones en red, almacenamiento en iCloud, contiene el sistema de respaldo de datos (Backup Core), codificación de texto, indexación de documentos y, en general, proporciona todas las API del sistema.

- Core OS

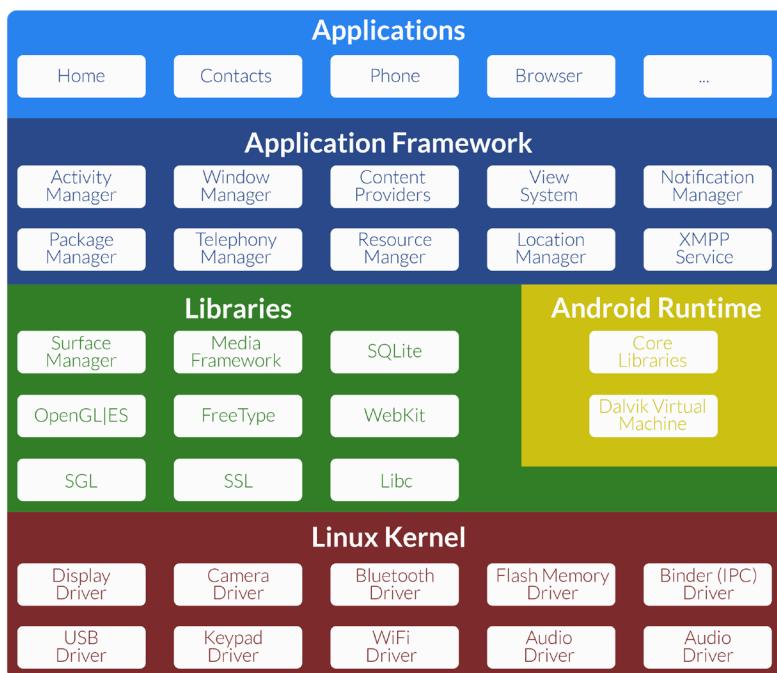
Componente principal del sistema. Representa el núcleo y proporciona gestión de seguridad, manejo de ficheros, administración de drivers, manejo de directivas de seguridad, direccionamiento de procesos en memoria, administra y controla procesos que acceden directamente al hardware.

Analicemos algunos datos importantes sobre este sistema operativo:

1. Hasta el 2009, el sistema operativo recibe el nombre de Iphone OS; a partir de 2010, es nombrado como iOS (Wikipedia, 2019).
2. Desde la versión 10.3.x se utiliza el sistema archivos APFS que reemplaza al HFS+; este nuevo sistema de archivos fue desarrollado específicamente para arquitectura de 64 bits.

3. Desde la versión iOS4 se implementa el segundo plano en el escritorio del sistema con un manejo eficiente de multitarea y animaciones; la herramienta que posibilita esto es conocida como UIKit la cual permite administrar la interacción del usuario con el sistema y gestionar eventos. Referencia disponible en <https://developer.apple.com/documentation/uikit>
4. La documentación del funcionamiento interno del sistema se encuentra en Apple Developer Documentation. (2019).
5. **Se tiene un lenguaje de programación adicional para la creación de aplicaciones móviles para Apple:** Swift, el cual incorpora cambios en la sintaxis, manejo de variables, clases, funciones y operadores, removiendo el uso de apuntadores; las líneas de código necesarias para la implementación de métodos es menor, dada la simplificación de sintaxis (en comparación con Objective-C) (González García, Pascual Espada, Pelayo G-Bustelo, & Cueva Lovelle, 2015).
6. Las herramientas de desarrollo para crear aplicaciones nativas en este sistema operativo se encuentran en el iOS SDK, disponible en <https://developer.apple.com/>.





En la imagen podemos observar la arquitectura del sistema operativo Android, un modelo en cuatro capas.

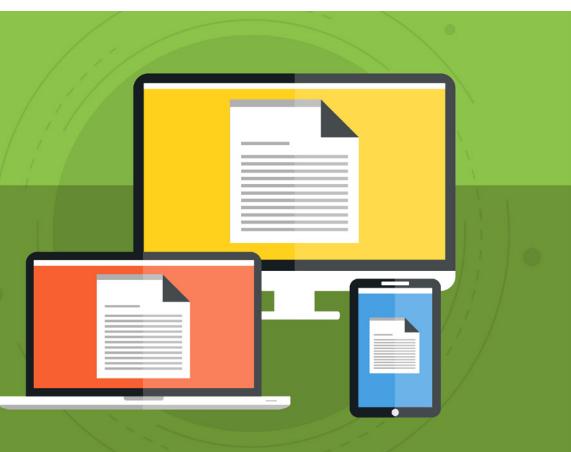
Esas capas cumplen las siguientes funcionalidades:

- Kernel de Linux. Ofrece los drivers necesarios para el manejo del hardware; se encuentran entonces los controladores de dispositivo. Proporciona servicios de seguridad, manejo de memoria, ofrece soporte a multiproceso.
- Librerías. Cuenta con librerías nativas, se trata de funciones y clases para manejo de bases de datos (SQLite), gestión de gráficos y multimedia, manejo de fuentes tipográficas y protocolo de conexión segura a internet. En esta capa se encuentran instrucciones escritas en C y C++ que representan las bibliotecas nativas del sistema.
- Runtime de Android. Gestión de procesos fundamentado en máquina virtual; permite ejecutar aplicaciones programadas en Java con gestión eficiente de consumo de energía. El primer entorno de máquina virtual es Dalvik y a partir de la versión 5, es reemplazado por ART (Android Runtime); respectivamente, la compilación de aplicaciones se da por los modelos JIT (just-in-time) y AOT (ahead-of-time) Android (2019).
- Entorno de aplicaciones (Application framework). Proporciona la plataforma de desarrollo con manejo de sensores, notificaciones, gestión de ventanas e interfaces de usuario.
- Aplicaciones. Programas predeterminados del sistema y adicionados por el usuario.

Analicemos dos datos importantes sobre Android:

Los sistemas de archivos disponibles en Android son Ext4 y F2FS.

El entorno de desarrollo nativo es Android Studio, con soporte a diferentes API, de acuerdo con la versión a la que se enfocará la aplicación. Los lenguajes de programación disponibles son Java, Kotlin y Dart.



Hablemos ahora sobre los teléfonos inteligentes o *SmartPhones*. Estos se aproximan a las prestaciones de una estación de cómputo de mesa o portátil. En la actualidad, la brecha se difumina tanto en hardware como en software; las prestaciones se incrementan y el desarrollo de software cuenta con múltiples opciones y entornos.

En este sentido, independiente del sistema operativo, es posible crear aplicaciones sin diferenciar el sistema operativo y un mismo código es válido para Android, iOS, entre otros, con su respectivo lenguaje de programación. Se tienen disponibles:

- React Native: JavaScript
- Ionic Framework: JavaScript, TypeScript
- Adobe PhoneGap: JavaScript
- Xamarin: C#

Independiente de la plataforma, sistema operativo o lenguaje de programación, el desarrollo de aplicaciones se da en capas de nivel superior, donde se encuentra el framework de desarrollo.

Los tipos de aplicaciones pueden ser, según Castillo & Sigel (2018):

- Web móviles

Se despliegan en un navegador web, usando HTML5, CSS3 y JAVA, con la ventaja que cualquier tipo de dispositivo visualiza y despliega la aplicación, independiente del sistema operativo. El tipo más representativo en este modelo son las Aplicaciones Web Progresivas (PWA). No requieren instalación en el dispositivo móvil y su interfaz es similar a una aplicación nativa. El diseño de la aplicación se adapta al tamaño del dispositivo que la usa y esta característica es conocida como diseño adaptativo o Responsive Web Design. No utilizan un SDK para su desarrollo y no cuentan con gestión de recursos como memoria, sensores u otros recursos propios del hardware del dispositivo móvil.

- Híbridas-web

Lleva una aplicación web empaquetada en una aplicación nativa, instalable o residente en el sistema, donde el navegador está embebido o integrado en la aplicación. Al igual que las aplicaciones web móviles, hacen uso del principio WORA, un mismo código se compila para múltiples plataformas. Algunos de los entornos que implementan este principio son: Apache Cordova, Adobe PhoneGap, Ionic Framework, Apache Weex.

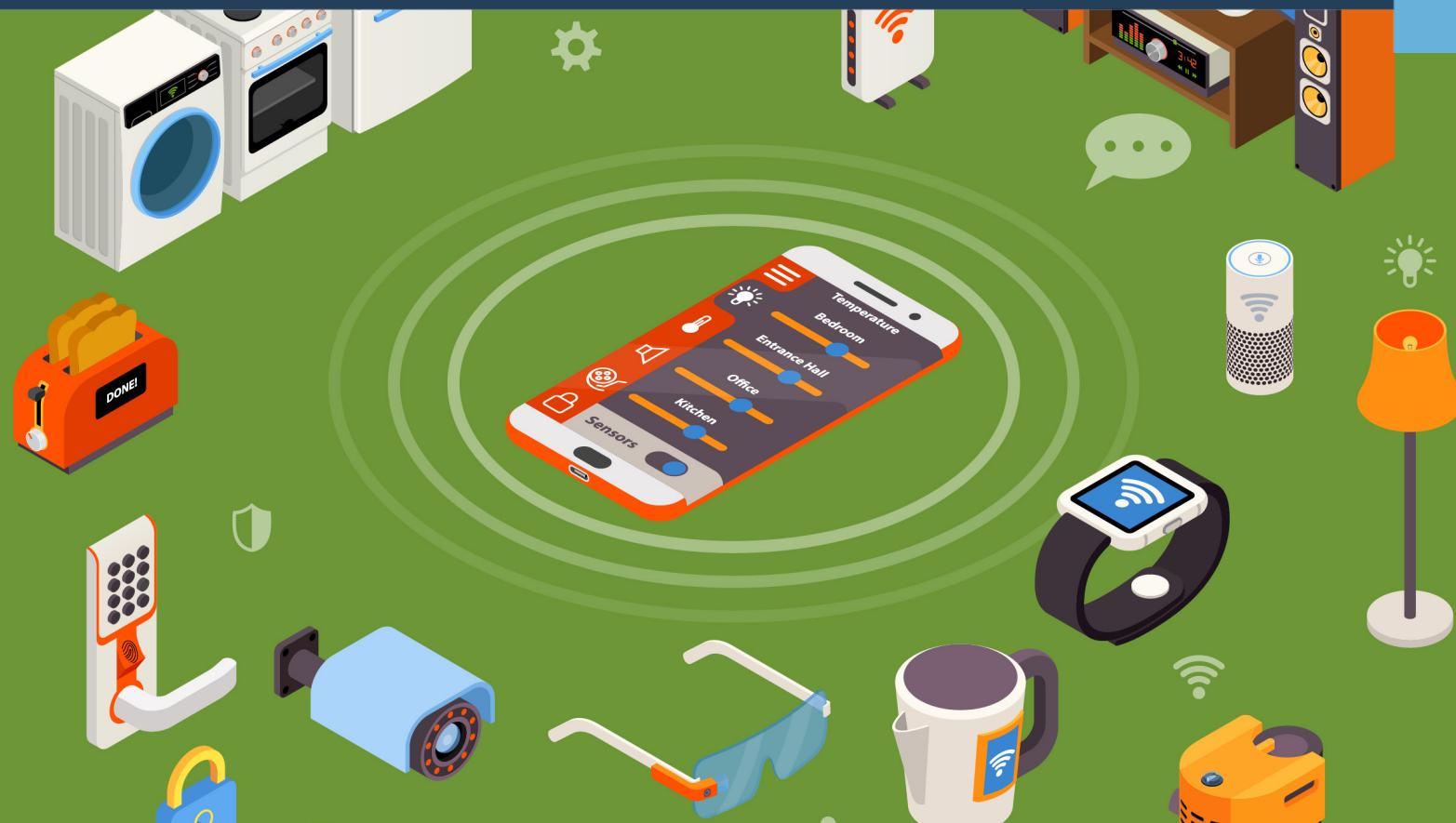
- Mixtas

Combina técnicas de programación web y nativas en una aplicación.

- Nativas

Aplicaciones que se encuentran instaladas en el sistema de archivos de un dispositivo, utilizan tecnología propia de cada sistema operativo, aprovechan todas las funcionalidades del dispositivo móvil y acceden o controlan el hardware. Se distribuyen en los canales oficiales de reparto de cada proveedor: App Store o Google Play Store.

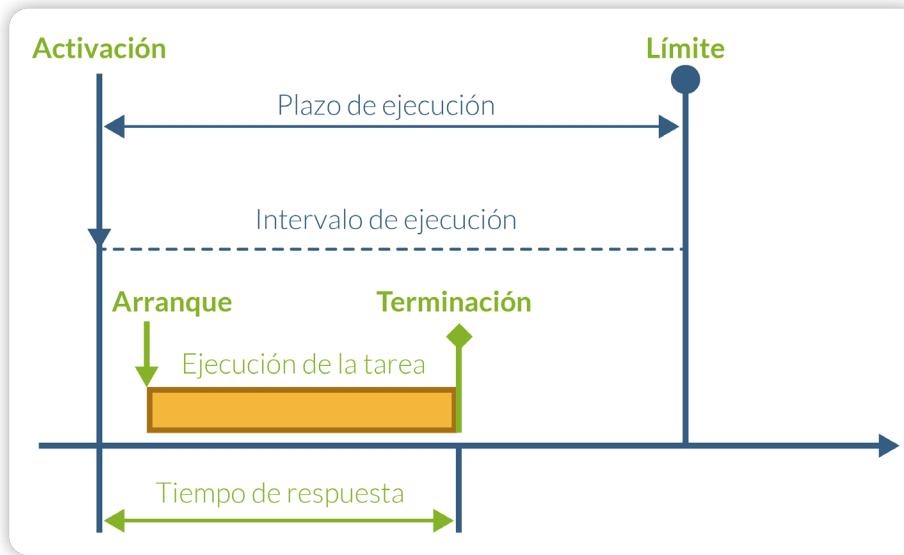
La instalación de los sistemas operativos móviles depende del fabricante del hardware, a diferencia de un equipo de escritorio o un equipo portátil; el sistema operativo no se puede instalar o actualizar libremente, debido a que las API limitan la compatibilidad.



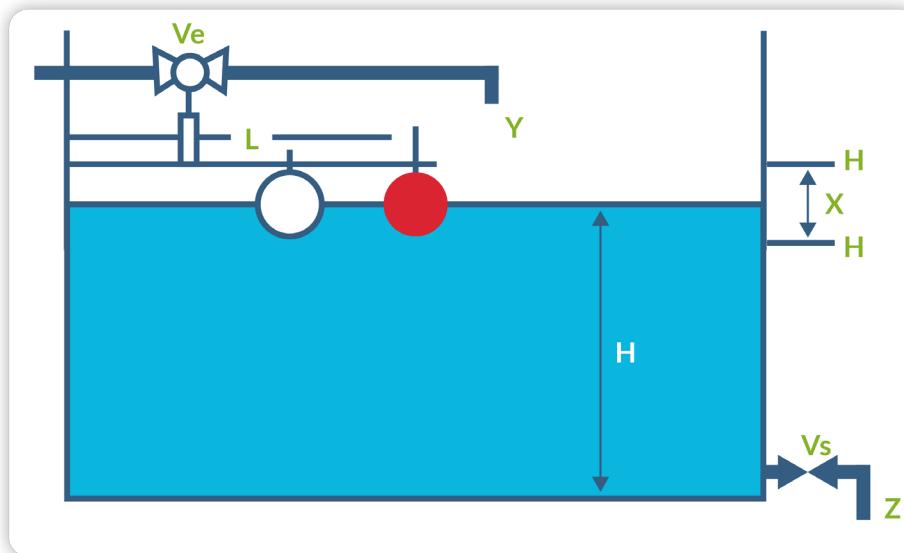
2 Tema

Sistemas operativos en tiempo real (RTOS o SOTR)

Dentro de los tipos de sistemas operativos, encontramos aquellos en tiempo real, los cuales se caracterizan por controlar o monitorear información de entrada y salida en un contexto o entorno dado, bajo unas restricciones de tiempo específicas. Tal es el caso de sistemas industrializados, donde se deben llevar a cabo acciones específicas en una cadena de producción y reaccionar ante los eventos del exterior registrados constantemente por el sistema (Tanenbaum, 2003).



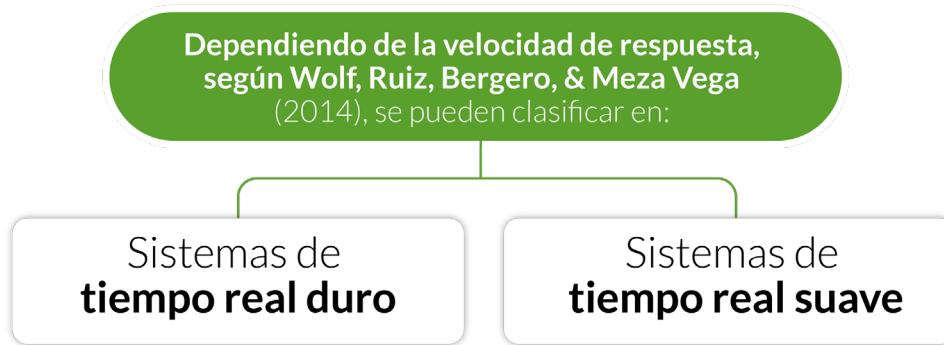
Se utilizan primordialmente en sistemas embebidos o de microcontroladores para automatización de procesos físicos, líneas de producción o fabricación, robótica, o internet de las cosas. Típicamente se tienen componentes electrónicos y mecánicos trabajando en un entorno con cambios constantes (procesos con naturaleza cambiante); se tienen diferentes sensores y actuadores que controlan el comportamiento de un entorno determinado, la información con la que interactúa el sistema, bien sea entrada o salida; puede ser análoga o digital, es decir, puede tener valores continuos o discretos.



La imagen nos muestra un caso de aplicación, donde se tiene una válvula (Ve) que se encarga de habilitar o cerrar el flujo de agua en un tanque y otra válvula (Vs) de salida o evacuación de agua; un sistema operativo, en tiempo real, se encarga en este caso de monitorear o medir el

nivel actual del tanque (H) y dado un nivel de umbral o un tope mínimo preestablecido, decide si debe abrir o cerrar la válvula de entrada o salida, según el caso, para mantener el nivel adecuado previamente definido.

Estos sistemas operativos pueden formar parte de un sistema mayor, altamente integrado y dedicado a realizar una función determinada.



Dependiendo de la velocidad de respuesta, según Wolf, Ruiz, Bergero, & Meza Vega (2014), se pueden clasificar en:

- Sistemas de tiempo real duro. Se garantiza que un proceso se finaliza en un tiempo determinado o antes, buscando reducir los retardos en entrada, salida y procesamiento al máximo, de manera que el funcionamiento del sistema en conjunto no se paralice o retrase. Este es el caso de sistematización o asistencia de procesos médicos, seguridad y demás sistemas de control que deben cumplir con restricciones temporales para ejecución de tareas críticas o vitales.
- Sistemas de tiempo real suave. Existe un margen de tolerancia a fallos o respuestas que sobrepasan el tiempo establecido para un sistema, funcionalidad dada en la mayoría de sistemas operativos, sistemas de comunicación en red.

De acuerdo a Garibaldi (2009), en los SOTR encontramos las siguientes características fundamentales:

- Determinismo. El tiempo de respuesta está considerado en un intervalo fijo o preestablecido; para una tarea de alta prioridad, la fluctuación de retardo tiende a cero.
- Sensibilidad. Una vez identificado un nuevo evento en el sistema, como el cambio de estado o nuevo valor en un sensor, el sistema gestiona prioridades e inicia el proceso determinado. Junto con el determinismo, esta característica establece la respuesta a eventos externos. Si un proceso de mayor prioridad se inicia, el sistema gestiona los recursos necesarios para llevar a cabo la tarea en su nivel de importancia determinado.

- Control del usuario. Provee una interfaz al usuario para que pueda establecer o decidir la prioridad para los eventos.
- Fiabilidad. Se garantiza un rendimiento en la respuesta a estímulos externos recibidos por sensores; las pérdidas o degradaciones de información tienden a ser mínimas.
- Tolerancia a fallos. Conserva la mayor cantidad posible de datos ante un fallo general en el sistema, sin comprometer el funcionamiento íntegro. Se busca minimizar efectos adversos ante una falla, de manera que los procesos se mantengan en línea, desde el nivel más alto de prioridad.

Teniendo claro el concepto de SOTR, estudiemos algunos ejemplos.

- Integrity

Es usado en sistemas de comunicación y navegación aérea, telecomunicaciones, aplicaciones médicas, producción de automóviles, entre otros campos industriales.

Si deseas conocer más detalles técnicos, visita el siguiente enlace: <https://www.ghs.com/products/rtos/integrity.html>

- freeRTOS

Implementado en el lenguaje de programación C, es uno de los más usados en sistema embebidos. Permite gestionar y priorizar múltiples procesos o hilos. Uno de los estándares más usados que define la gestión de hilos y el desarrollo de aplicaciones en tiempo real es POSIX.4, el cual está implementado en este sistema operativo. Se puede usar en plataformas como Arduino y Raspberry.

Si deseas conocer más sobre la documentación de este sistema, visita el siguiente enlace: https://www.freertos.org/Documentation/RTOS_book.html

- SafeRTOS

Es un sistema basado en freeRTOS, y cumple con los requisitos de seguridad funcional de sistemas de control IEC61508, la cual contempla la normalización de seguridad en el diseño de sistemas eléctricos y electrónicos.

Si deseas conocer más sobre las hojas de características, visita el siguiente enlace: <https://www.highintegritysystems.com/safertos/>

- Windows IoT Core

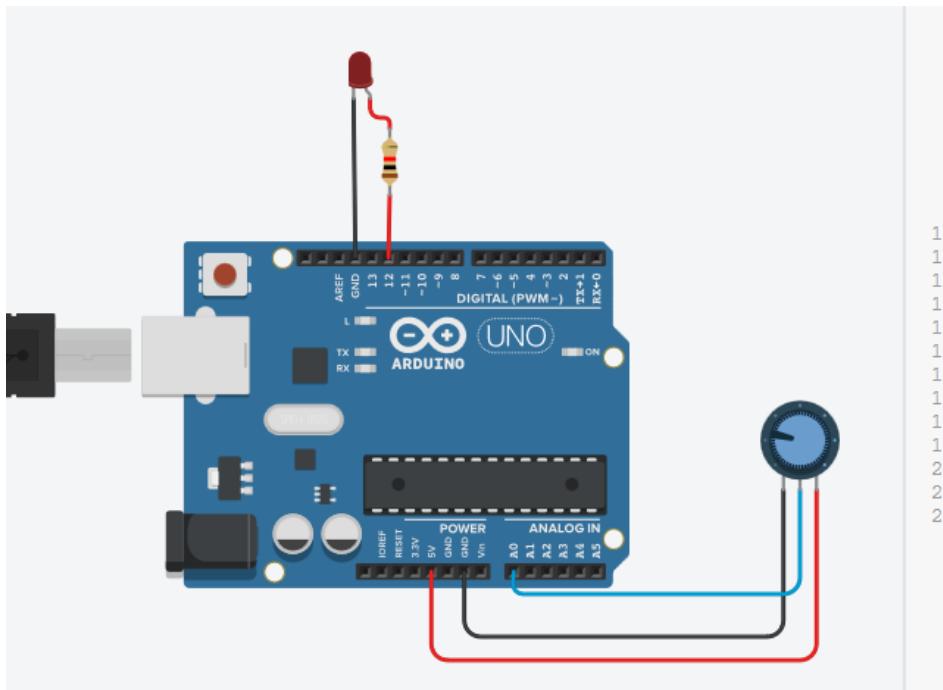
Es un sistema de 32 bits desarrollado por Microsoft específicamente para procesadores ARM, con funcionalidades aplicables al internet de las cosas. Su versión diseñada para sistemas embebidos es Windows Embedded.

Si deseas conocer una descripción detallada, visita el siguiente enlace:
<https://developer.microsoft.com/es-es/windows/iot>

- VxWorks

Es un sistema basado en Unix, con compatibilidad POSIX, con respuesta dada en nanosegundos, ampliamente implementado en sistemas embebidos.

Si deseas conocer sus beneficios y características, visita el siguiente enlace:
<https://www.windriver.com/products/vxworks/>



Los lenguajes de programación más implementados en el desarrollo de aplicaciones para este sistema son: C, Java, Ada. La imagen de arriba muestra la estructura general de un código implementado en C para Arduino.

Aquí se aprecian los bloques típicos de estas aplicaciones: un bloque de definición de variables (previo al bloque `setup`), establecimiento de entradas y salidas (`setup`), y un ciclo infinito (`loop`), que monitorea constantemente las entradas para llevar a cabo las acciones necesarias.

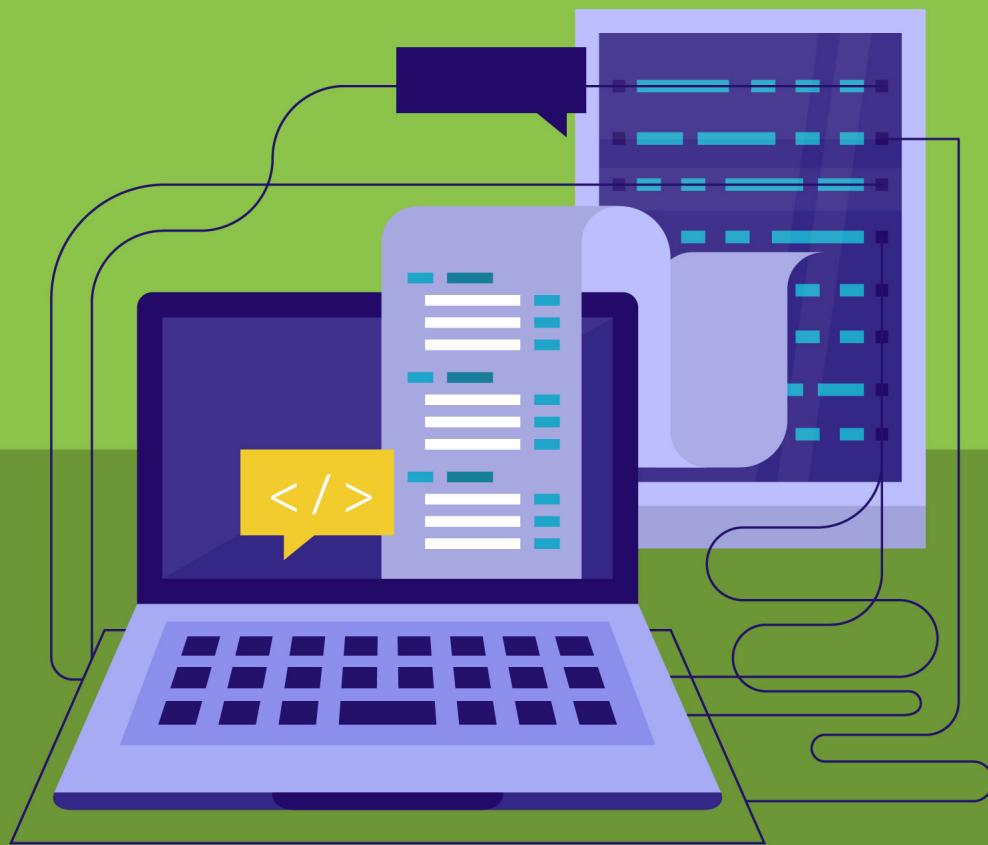
En este caso, se definen los componentes a utilizar; conectados en los pines A0, entrada en la parte inferior (línea 2), y el pin 12, salida donde se tiene unido un diodo led emisor de luz en la parte superior (línea 1). Se define una variable donde se va a recibir el estado de la entrada (variable val inicializado en cero en la línea 3). El bloque setup define la salida respectiva.

En el bloque loop se monitorea constantemente el valor de la entrada que puede cambiar su valor continuo a elección del usuario, dependiendo del valor registrado; si está por debajo de 255, la salida se lleva a cero, es decir, se apaga; de lo contrario, se gradúa su valor a un cuarto del total del valor registrado en la entrada, o sea, se enciende el diodo led con una intensidad variable y correspondiente.

El trabajo en tiempo real para Arduino se obtiene al agregar la siguiente línea al inicio del proyecto, adicionando la librería para crear tareas paralelas y administradas por el sistema:

```
#include <Arduino_FreeRTOS.h>
```





Tema **3**

Sistemas operativos industriales

Los sistemas operativos permiten:

Implementar complejos algoritmos de control adaptativo y optimizado en contextos de redes.

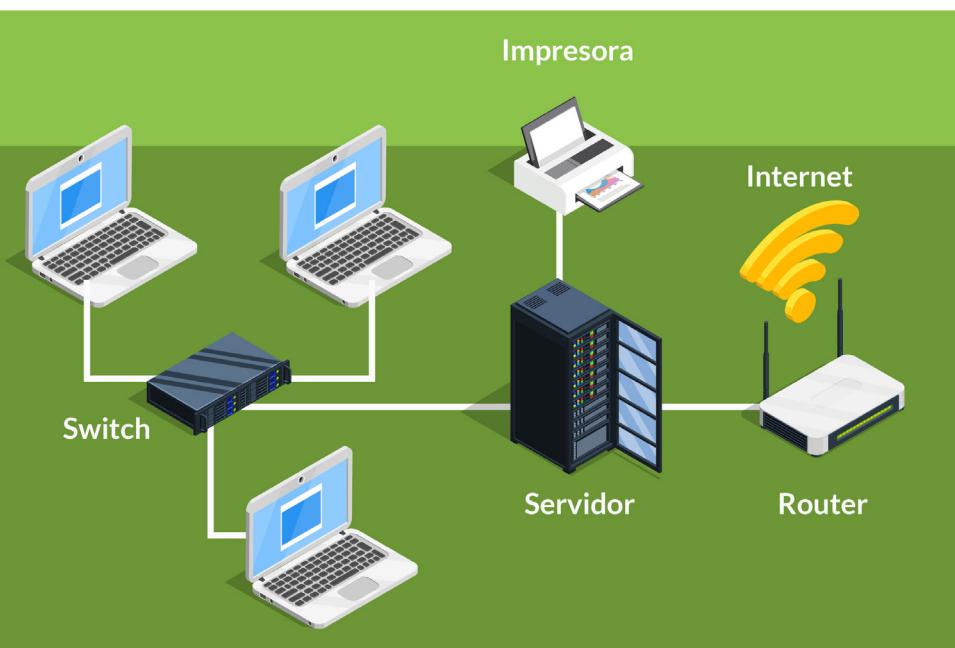
Realizar un control integral en plantas de fabricación.

Múltiples procesos de producción y procesamiento de datos.

También buscan incrementar la eficacia en la ejecución de operaciones, dotar de mayor seguridad el proceso, reducir considerablemente las operaciones manuales y facilitar tareas de automatización y control.

Las características de sistemas operativos con aplicaciones industriales son:

- Gestión de datos. Monitoreo, procesamiento, recolección o análisis de datos con alto nivel de confidencialidad e integridad, bien sea en tiempo real o almacenamiento permanente. Pueden alimentar el sistema para toma de decisiones, analítica, y generación de informes. Se garantiza la disponibilidad de la información a pares de componentes autorizados en el sistema.
- Supervisión. El sistema se integra o conecta a diferentes actuadores, reguladores o estaciones de trabajo donde se informa del estado actual, rendimiento o ubicación de cada componente del sistema. Se suele tener un panel de visualización con los diferentes elementos conectados al sistema.
- Control. Puede realizar acciones secuenciales o condicionales dadas las mediciones o datos tomados. Se pueden dar a componentes mecánicos, neumáticos, eléctricos, requiriendo acoplos o conversiones análogos – digital. Requieren componentes adicionales y externos de hardware. EL proceso se gestiona por procesos de lotes, planificación de prioridad y modelos de colas (Tanenbaum, 2003).



Los sistemas operativos industriales comparten características y funcionalidades con los sistemas operativos en tiempo real, pero los diferencia la complejidad, escala o volumen de componentes que pueden administrar simultáneamente, requiriendo mayor poder de procesamiento y almacenamiento.

Estas funcionalidades se dan sobre servidores, los cuales se comportan como distribuidores de servicios o proveedores de recursos.

El sistema operativo de Microsoft, para este enfoque, es Windows Server, el cual trabaja en un dominio identificado con un nombre único, que representa un conjunto de estaciones de trabajo (servidores y clientes) que comparten base de datos que contiene las credenciales y permisos de todos los usuarios con sus respectivos registros de actividades y políticas de

seguridad. El sistema puede compartir diferentes recursos en el dominio como impresoras, carpetas y archivos, entre otros, de forma centralizada con restricciones de seguridad definidas por el administrador del sistema. Maneja múltiples protocolos de comunicación que garantizan la compatibilidad con otros sistemas operativos. (Raya Cabrera & Santos González, 2014). El sistema de archivos es NTFS.

Los componentes más importantes en este sistema son:

- Directorio activo (Active Directory). Gestiona la información de usuarios y acceso a recursos diferenciados por roles.
- Servicio de archivos. Gestiona la indexación de información en el servidor y su respectivo acceso por usuarios del sistema.
- Servidor de protocolo de configuración dinámica de host (DHCP). Asigna a los equipos o dispositivos conectados a la misma red una dirección IP que los identifica de forma única.
- Servidor DNS. Traduce o asocia direcciones IP a nombres de dominio.
- Servidor web. Permite alojar una página web (tal como lo hace un servicio de hosting en la nube) a la cual se puede acceder desde un navegador web en la red local. Se denomina Internet Information Services (IIS) y es compatible con la tecnología de ASP.NET.

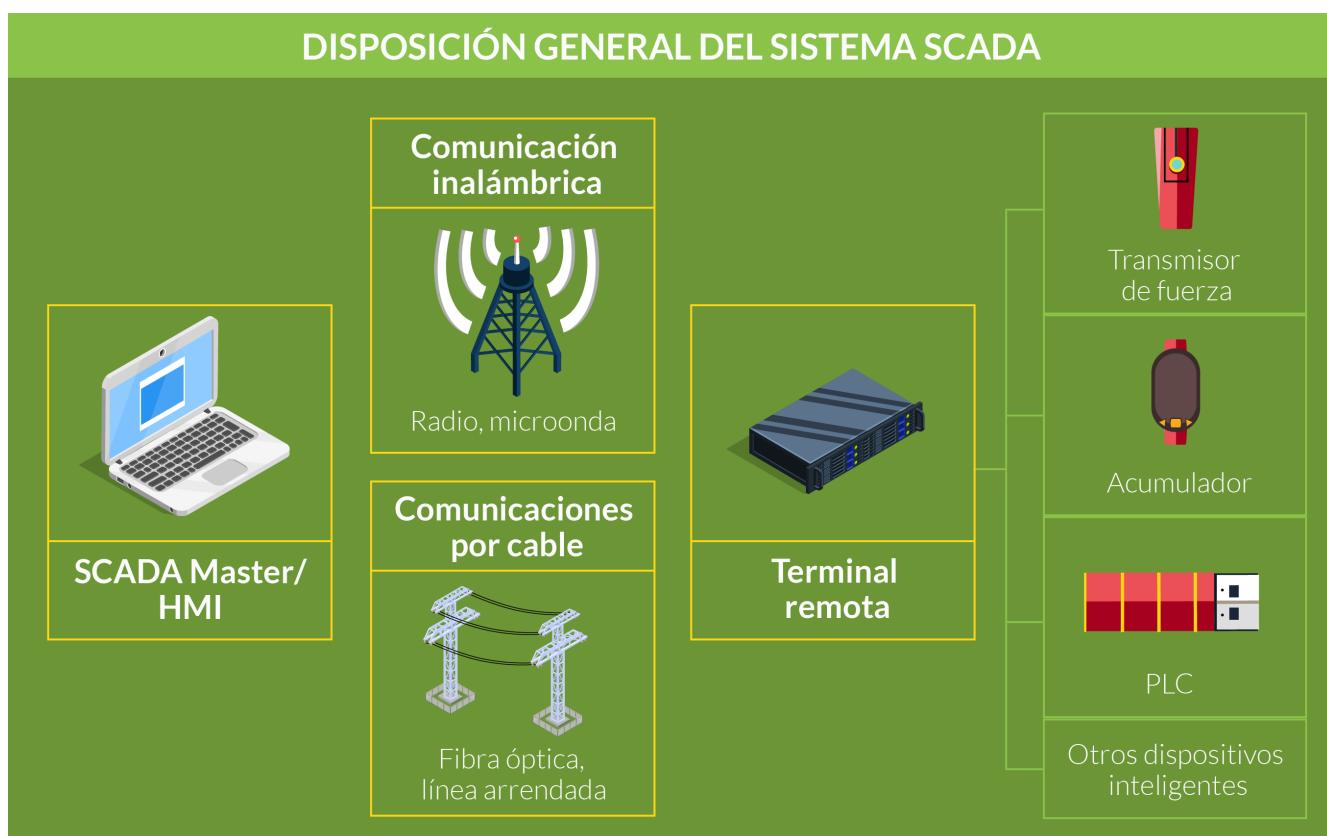
Desde el punto de vista UNIX, la implementación en servidores ha sido el punto fuerte de Linux y su kernel se ha especializado en el trabajo distribuido en redes. Las versiones más representativas son RedHat y SUSE (Raya Cabrera & Santos González, 2014).

Por parte de RedHat, se cuenta con RHEL (Red Hat Enterprise Linux) y sus características son: (<https://www.redhat.com/es/technologies/linux-platforms/articles/red-hat-enterprise-linux-5-features-and-benefits>):

- Administración de sistema. Condensa la configuración de todo el sistema en la interfaz Cockpit, lo que integra: gestión de cuentas de usuarios, configuración de parámetros de red, monitorio y configuración de recursos y servicios.
- Disposición de contenedores. Permite proporcionar de forma aislada aplicaciones en entornos de producción, desarrollo o prueba con alto nivel de seguridad.
- Almacenamiento. Cuenta con soportes para sistemas de archivos Ext4 y utiliza uno optimizado para 64 bits: XFS. Dispone de sistemas para copia de seguridad.
- Base de datos. Gestión realizada con el sistema MariaDB como implementación de MYSQL.

- Servidor web. Alberga sitios web por medio de servidor Web Apache compatible con la tecnología de PHP.
- Interoperabilidad. De esta distribución RHEL, se ha derivado CentOS que gana mayor terreno en el mercado de servidores (Membrey, Verhoeven, & Angenendt, 2009) y la encuesta de w3techs (2019) demuestra que los sistemas Linux son mayoritariamente implementados en sitios web y la tendencia va en alza.

Su sistema de comunicación es compatible con UNIX y Microsoft Windows Server.



Pero dadas las necesidades de control de procesos en zonas de cobertura extendidas, particularmente para el sector industrial, se crean sistemas supervisores de adquisición y control de datos (SCADA). A continuación, se describen sus principales componentes según Corrales Paucar (2007):

- SCADA Master

Sistema principal o estación central con un sistema operativo servidor que permite supervisar un proceso determinado, se denomina también unidad terminal maestra, MTU. Despliega una interfaz que visualiza gráficamente las variables de un proceso, lo que se

conoce como Interfaz Hombre Máquina (HMI) y permite interactuar plenamente con el proceso, identificar sus medidas o accionar actuadores.

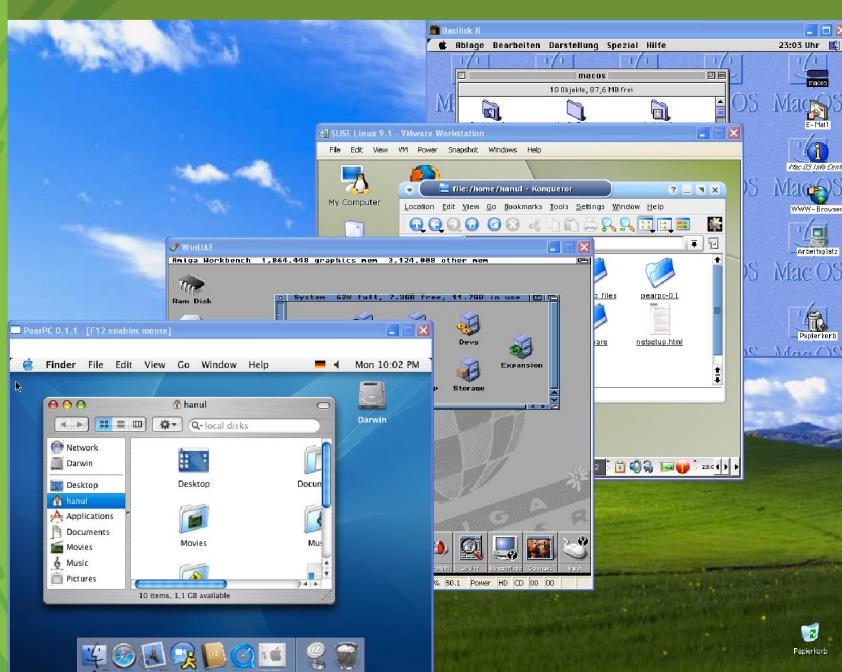
- Terminal remoto

Dispositivos o periféricos que permiten el control o la adquisición de datos. Son llamados unidades de telemetría remota (RTU) o dispositivos de campo y decodifica información de transductores, sensores, actuadores desde y hacia la estación central. Normalmente integran un sistema operativo de tiempo real. Los más utilizados son controladores lógicos programables (PLC).

Recuerda que...

La sistematización permite una supervisión especializada de un proceso que incluye automatización de tareas, lectura de datos, diagnóstico y detección de fallas, generación de alarmas en un modelo cliente – servidor, para sectores industriales o manufactureros, con aplicaciones en domótica, sector salud, aeronáutica, fabricación automotriz y demás sectores interesados en gestionar datos de fenómenos físicos o químicos, remotamente.

Fabricantes de hardware y software proponen herramientas de gestión, como es el caso de Andover Continuum de Schneider Electric. Inclusive la robótica cuenta con sistemas especializados, como el caso de Robot Operating System (ROS) y el entorno de Microsoft Robotics Developer Studio.



4 Tema

Sistemas operativos virtuales

¿Sabías que la virtualización permite desplegar servicios que usualmente se ofrecen en servidores físicos independientes, para optimizar el hardware disponible con un entorno simulado o virtual?

Entonces, el software encargado de administrar el hardware y dividir los entornos simulados, se denomina hipervisor (Wolf, Ruiz, Bergero, & Meza Vega, 2014) y origina las máquinas virtuales.

El propósito principal es entregar recursos a usuarios, en un determinado contexto, para crear múltiples entornos desde un solo sistema de hardware. Este enfoque ha originado la computación en la nube (Cloud computing), donde se comparten por medio de la virtualización, servicios y recursos dispuestos en la red.

Herramientas disponibles

Para la virtualización se tienen, entre otras, las siguientes herramientas disponibles:



VirtualBox

- **VirtualBox.** Desarrollado por Oracle, permite a un sistema operativo base, llamado anfitrión, instalar otros sistemas operativos, denominados invitados. Al momento de instalar un nuevo sistema operativo, permite seleccionar los recursos que va a demandar el anfitrión: memoria RAM, procesadores, video, red.



VMware

- **VMware.** Es un software propietario de Dell que, adicionalmente a la virtualización de sistemas operativos, ofrece la virtualización de escritorio (VMware Horizon), donde una estación remota utiliza los recursos del servidor.



Microsoft Hyper-V Server

- **Microsoft Hyper-V Server.** Se encuentra disponible desde Windows Server 2008 y trabaja en sistemas de 64 bits. Soporta arquitectura de multiprocesamiento simétrico (SMP) y permite la virtualización de redes con HNV, característica introducida desde Windows Server 2012, que permite a un nodo trabajar en una infraestructura de red compartida y virtualizada.



Vagrant

- **Vagrant.** Es una herramienta que permite crear entornos de desarrollo de software con características determinadas, basado en un sistema virtualizado, que se integra a VirtualBox o Vmware.

Computación en la nube

Desde el punto de vista de la computación en la nube, se tiene acceso a múltiples servicios, aplicaciones, espacio de almacenamiento y cantidad de funcionalidades, donde las principales alternativas, según Aguilar (2009), son:



Amazon Web Services

- Amazon Web Services (AWS). Es una plataforma que contiene un conglomerado de servicios de virtualización en la nube, contando con múltiples sistemas de gestión de bases de datos, almacenamiento o hosting, procesamiento, aprendizaje automatizado, entre otros.

Si deseas conocer su listado de productos, copia en tu navegador el siguiente enlace:

<https://aws.amazon.com/es/>



Google Cloud Platform

Google Cloud Platform

- Google Cloud Platform. Ofrece soluciones de almacenamiento, desarrollo de aplicaciones, procesamiento, analítica de datos y demás.

Si deseas conocer sus productos y servicios detallados, haz clic en el siguiente enlace:

<https://cloud.google.com/products/>



Microsoft Azure

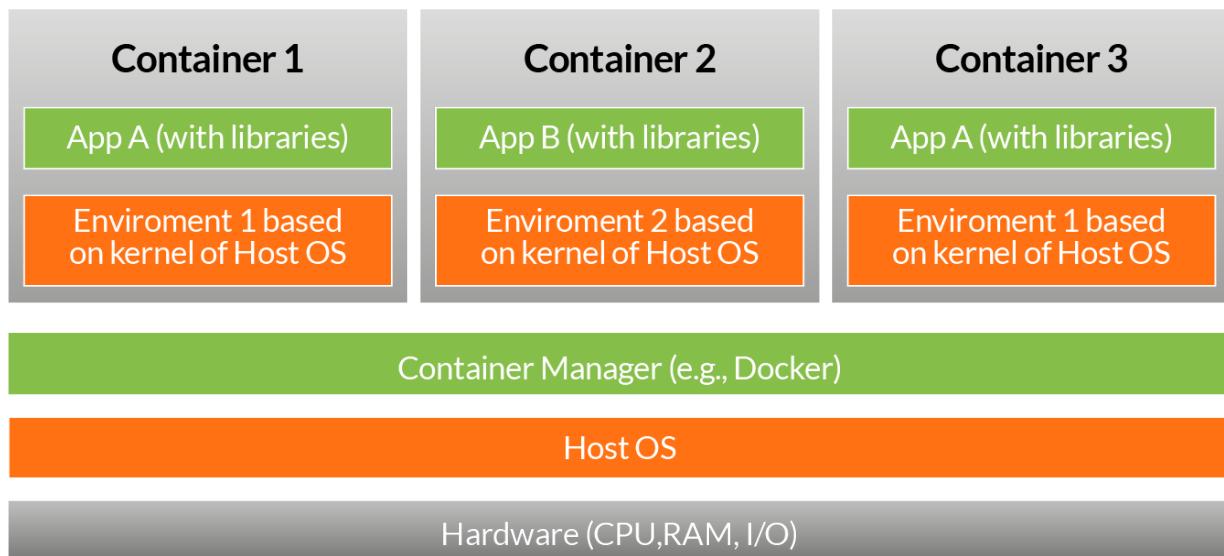
- Microsoft Azure. Sistema que permite administrar y desplegar recursos de almacenamiento y procesamiento en línea, donde se destacan bases de datos, máquinas virtuales y entorno de desarrollo, entre otros.

Si deseas conocer un compendio completo de sus servicios, visita la siguiente dirección:

<https://azure.microsoft.com/es-es/services/>

Virtualización basada en contenedores

Se trata de otro modelo de virtualización, donde no se crea una máquina virtual completa (el nuevo sistema operativo no se instala de forma independiente íntegramente), sino que se virtualiza el sistema operativo anfitrión, donde se crean entornos independientes que aprovechan el mismo núcleo (Domínguez San Segundo, 2016). Una de las herramientas más implementadas es docker, que ofrece un entorno de desarrollo y producción de software óptimo (Pacheco Laje, 2018).



Puedes observar su modelo en la imagen, donde se aprecia un único sistema operativo (anfitrión o Host OS) y múltiples entornos o contenedores, con sus respectivas aplicaciones y librerías.



Esta licencia permite a otros distribuir, remezclar, retocar, y crear a partir de esta obra de manera no comercial y, a pesar que sus nuevas obras deben siempre mencionar a la IU Digital y mantenerse sin fines comerciales, no están obligados a licenciar obras derivadas bajo las mismas condiciones.

