Tarea 3. Conceptos de software y antecedentes de los sistemas distribuidos

SISTEMAS CONCURRENTES Y DISTRIBUIDOS

Ignacio David Vázquez Pérez

218292866

Tarea 3. Conceptos de software y antecedentes de los sistemas distribuidos

Objetivo:

Comprender los conceptos fundamentales del software: Esto incluye la definición de software. Entender los sistemas distribuidos. Esto implica aprender sobre la definición de los sistemas distribuidos, sus características como la concurrencia, la carencia de un reloj global y los fallos independientes de los componentes.

Introducción:

Exploraremos la Taxonomía de Flynn, una clasificación fundamental de las arquitecturas de computadoras según la forma en que los procesadores manejan las instrucciones y los datos. Además, examinaremos cómo el software puede organizarse en la CPU, desde sistemas monolíticos hasta microkernels y máquinas virtuales. Estos conceptos son esenciales para cualquier estudiante o profesional de la informática, ya que sientan las bases para el diseño y la optimización de sistemas informáticos modernos. Estos temas proporcionan una base sólida para entender cómo las computadoras procesan datos y ejecutan programas de software de manera eficiente.

Actividades a realizar

Si el software es amorfo, ¿cómo se puede clasificar?

Investiga cómo se puede clasificar y ejemplifica los términos.

1. **Por Funcionalidad:**
    - **Software de sistema:** Este tipo de software está diseñado para gestionar y controlar los recursos de hardware de una computadora. Ejemplos incluyen sistemas operativos, controladores de dispositivo y utilidades de sistema.

- **Software de aplicación:** Son programas diseñados para realizar tareas específicas para los usuarios finales. Ejemplos incluyen procesadores de texto, hojas de cálculo, navegadores web y software de diseño gráfico.

2. **Por Licencia:**
   - **Software propietario:** Este tipo de software está protegido por derechos de autor y no se puede modificar ni redistribuir libremente. Los usuarios suelen tener que pagar una licencia para su uso.
   - **Software de código abierto:** El software de código abierto permite a los usuarios acceder al código fuente, modificarlo y redistribuirlo libremente. Ejemplos incluyen Linux, Mozilla Firefox y Apache.
3. **Por Plataforma:**
   - **Software de plataforma cruzada:** Puede ejecutarse en múltiples sistemas operativos o arquitecturas de hardware. Ejemplos incluyen Java y aplicaciones web.
   - **Software específico de plataforma:** Diseñado para ejecutarse en una plataforma o sistema operativo específico. Ejemplos incluyen software exclusivo para Windows o macOS.
4. **Por Estructura:**
   - **Software monolítico:** El software monolítico es una aplicación en la que todas las funciones y componentes se integran en una única unidad ejecutable. No se separa en módulos independientes. Ejemplo: sistemas operativos tradicionales como Windows.
   - **Software basado en microkernel:** Divide las funciones del sistema operativo en módulos independientes llamados microkernels. Ejemplo: el sistema operativo MINIX.
   - **Máquinas Virtuales:** El software de máquina virtual permite que una computadora ejecute múltiples sistemas operativos simultáneamente como máquinas virtuales. Ejemplo: VirtualBox.
5. **Por Sector de Uso:**
   - **Software empresarial:** Diseñado para empresas y organizaciones para satisfacer necesidades comerciales específicas, como sistemas de gestión empresarial (ERP) y software de contabilidad.
   - **Software de entretenimiento:** Creado para el entretenimiento, como videojuegos y aplicaciones multimedia.

Taxonomía de flynn y la clasificación del software

1. **SISD (Single Instruction, Single Data):**
   - **Acoplamiento en Hardware:** Fuertemente acoplado, ya que hay un solo procesador ejecutando una única instrucción y operando en un único conjunto de datos en un momento dado.
   - **Acoplamiento en Software:** El sistema operativo y las aplicaciones se ejecutan en un solo procesador.

- **Componentes:** Un solo procesador, memoria, periféricos y sistema operativo.
- **Principal Característica:** Eficiencia en aplicaciones de un solo hilo, no hay paralelismo de instrucciones ni de datos.

2. **SIMD (Single Instruction, Multiple Data):**
   - **Acoplamiento en Hardware:** Fuertemente acoplado, ya que hay un único controlador de instrucciones que ejecuta la misma instrucción en múltiples unidades de datos simultáneamente.
   - **Acoplamiento en Software:** El sistema operativo y las aplicaciones están diseñados para aprovechar la ejecución simultánea de una instrucción en múltiples datos.
   - **Componentes:** Un solo procesador con múltiples unidades de datos (por ejemplo, un conjunto de vectores o una matriz de procesadores).
   - **Principal Característica:** Adecuado para aplicaciones altamente paralelas, como procesamiento de imágenes o simulaciones científicas.
3. **MISD (Multiple Instruction, Single Data):**
   - **Acoplamiento en Hardware:** Fuertemente acoplado, ya que múltiples procesadores ejecutan diferentes instrucciones en un solo conjunto de datos.
   - **Acoplamiento en Software:** El sistema operativo y las aplicaciones están diseñados para dividir la tarea en múltiples instrucciones ejecutadas en paralelo.
   - **Componentes:** Múltiples procesadores que operan en un solo conjunto de datos compartido.
   - **Principal Característica:** Poco común en sistemas prácticos, se utiliza principalmente en escenarios de investigación.
4. **MIMD (Multiple Instruction, Multiple Data):**
   - **Acoplamiento en Hardware:** Puede ser tanto fuertemente acoplado como débilmente acoplado, dependiendo de la implementación. Los procesadores ejecutan instrucciones diferentes en diferentes conjuntos de datos.
   - **Acoplamiento en Software:** El sistema operativo y las aplicaciones pueden aprovechar el paralelismo tanto a nivel de instrucciones como a nivel de datos.
   - **Componentes:** Múltiples procesadores independientes, cada uno con su propia memoria y conjunto de instrucciones.
   - **Principal Característica:** Adecuado para aplicaciones altamente paralelas y distribuidas, común en sistemas multiprocesadores y computación en clúster.

Aspectos de Diseño de Sistemas Distribuidos:

1. **Transparencia:**
   - **Qué es:** Transparencia se refiere a ocultar la complejidad y las diferencias en la ubicación y acceso de los recursos distribuidos a los usuarios y aplicaciones.

- **Aspectos Involucrados:** Transparencia de acceso, transparencia de ubicación, transparencia de replicación, transparencia de migración, etc.
- **Ejemplo:** Cuando un usuario accede a un archivo en una red distribuida, no necesita conocer la ubicación física del archivo o cuántas copias existen.

2. **Flexibilidad:**
   - **Qué es:** Flexibilidad se refiere a la capacidad del sistema distribuido para adaptarse a cambios en los recursos y las necesidades de los usuarios sin interrupciones significativas.
   - **Aspectos Involucrados:** Reconfigurabilidad, capacidad de adaptación, fácil integración de nuevos nodos.
   - **Ejemplo:** Agregar nuevos servidores a una granja web sin afectar el rendimiento del sistema existente.
3. **Confiabilidad:**
   - **Qué es:** Confiabilidad se refiere a la capacidad del sistema distribuido para funcionar de manera continua y sin fallas.
   - **Aspectos Involucrados:** Tolerancia a fallas, redundancia, recuperación de fallos.
   - **Ejemplo:** Un sistema de almacenamiento distribuido que replica datos en varios servidores para garantizar la disponibilidad incluso si un servidor falla.
4. **Desempeño:**
   - **Qué es:** Desempeño se refiere a la velocidad y la eficiencia con la que el sistema distribuido realiza tareas y responde a las solicitudes de los usuarios.
   - **Aspectos Involucrados:** Latencia, ancho de banda, optimización de recursos, escalabilidad.
   - **Ejemplo:** Asegurar que un sistema de procesamiento de transacciones distribuidas pueda manejar un alto volumen de transacciones en tiempo real.
5. **Escalabilidad:**
   - **Qué es:** Escalabilidad se refiere a la capacidad del sistema distribuido para crecer y adaptarse a un aumento en la carga de trabajo o el número de usuarios.
   - **Aspectos Involucrados:** Escalabilidad horizontal (agregar más nodos), escalabilidad vertical (aumentar recursos en un nodo).
   - **Ejemplo:** Un sistema de redes sociales que puede manejar un aumento significativo en el número de usuarios sin degradación del rendimiento.

Conclusión

La Taxonomía de Flynn y la clasificación del software son dos conceptos fundamentales en el campo de la informática que nos ayudan a comprender cómo se organizan y operan los sistemas de computadoras y software. La Taxonomía de Flynn nos proporciona una clasificación esencial de las arquitecturas de computadoras en función de cómo los procesadores manejan las instrucciones y los

datos, dividiéndolos en cuatro categorías: SISD, SIMD, MISD y MIMD. Esta taxonomía es fundamental para comprender las capacidades de paralelismo y concurrencia de los sistemas de computadoras.

Por otro lado, la clasificación del software nos permite organizar programas y aplicaciones informáticas en función de sus características y usos específicos, como el software de sistema y el software de aplicación, el software propietario y el de código abierto, y muchos otros criterios. Esta clasificación es esencial para comprender cómo se desarrolla, implementa y utiliza el software en diversos entornos y aplicaciones.

Bibliografía

- Tanebaum Andrew. (1995). Sistemas Operativos Distribuidos. España. Prentice-Hall Hisp.
- Mclver Ann. (2011). Sistemas Operativos. México. Cengage Learning.
- Tanenbaum, A.,& Van Steen M.(2008). Sistemas Distribuidos, Principios y Paradigmas. (Segunda ed.). Prentice Hall.
- Tanenbaum, A.(2011). Redes de Computadoras. (Quinta ed.). Prentice Hall.
- Elmasri, R., Gil Carrick, A., & Levine, D. (2010). Sistemas Operativos, Un enfoque en espiral. McGraw--Hill.
- Universidad Virtual del Estado de Veracruz. (s. f.). Conceptos de software y antecedentes de los sistemas distribuidos. Cursos Clavijero. https://cursos.clavijero.edu.mx/cursos/179_is/modulo1/contenidos/tema1.1.html
- Sosa, V. J. (s. f.). Sistemas distribuidos: Panorama general. En Apuntes de la asignatura Sistemas Distribuidos (1a ed., Vol. 1). Tamaulipas: CINVESTAV. https://www.tamps.cinvestav.mx/~vjsosa/clases/sd/sistemas_distribuidos_panorama.pdf
- UNIR México. (2020). Ingeniería de software: qué es y cuáles son sus objetivos. UNIR México. https://mexico.unir.net/ingenieria/noticias/ingenieria-de-software-que-es-objetivos/