Sistemas Distribuidos
IF022

Curso 2021

Introduccion a los SD

Introduccion a los SD

- Definición

A distributed system as one in which hardware or software components located at networked computers communicate and coordinate their actions only by passing messages (Colouris)

A distributed system is a collection of autonomous computing elements that appears to its users as a single coherent system (van Steen - Tanenbaum)

Una colección de elementos computacionales autónomos (nodos) que se muestran a los usuarios cono un único sistema coherente.

Definición SD

- Aplicaciones en una computadora:
  - Shared Memory (IPC, Semáforos, Cola de Mensajes)

- vs Sistemas Distribuidos:
  - Ausencia de Shared Memory → intercambio de mensajes (acuerdos necesarios denominados *protocolos)*

- Problemas secundarios:
  - Concurrencia (acceso concurrente a recursos)
  - Inexistencia de Reloj Global (sincronización)
  - Fallas Independientes : los sd pueden fallas en nuevas formas

Definición SD

- Colección de nodos autónomos
  - Cada nodo es autónomo y de esta forma va a tener su propia noción del tiempo, no hay un reloj global. Esto lleva a problemas de coordinación y sincronización
  - Como manejar la pertenencia al grupo del SD (grupos abiertos y grupos cerrados group donde se necesita un mecanismo de admisión)

- Sistema coherente único
  - Un sistema distribuido es coherente si se comporta de acuerdo a lo que los usuarios esperan. La colección de nodos como un todo debe operar igual, sin importar donde, cuando y como la interacción entre los usuarios y el sistema tenga lugar

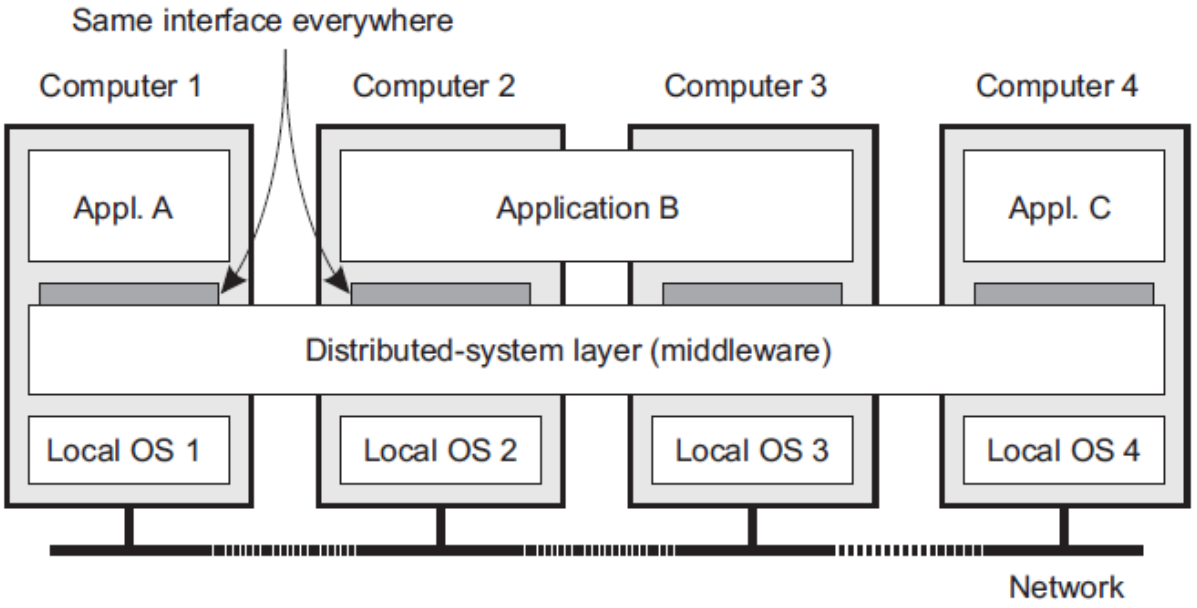
Examples

- An end user cannot tell where a computation is taking place
- Where data is exactly stored should be irrelevant to an application
- If or not data has been replicated is completely hidden

Keyword is distribution transparency

Middleware

- Es el Sistema Operativo de los sistemas distribuidos
  - Es una capa que se ubica por encima de los respectivos sistemas operativos de las computadoras que forman parte del SD
  - Contiene los componentes y funciones comunes que ya no será necesario implementar en las aplicaciones en forma separada
  - Provee los mecanismos para que una aplicación distribuida se comunique con otras y oculta las diferencias en hardware y sistemas operativos existentes entre los nodos



Middleware

- Maneja los recursos ocultando las particularidades a las aplicaciones cliente
    - Facilidades para comunicación entre aplicaciones
    - Servicios de seguridad
    - Servicios de registración
    - Enmascaramiento y recuperación ante fallas
- Servicios típicos del middleware
    - Comunicación (RPC: permite a una aplicación invocar una función que se implementa y se ejecuta en una computadora remota como si estuviera disponible localmente)
    - Transacciones: ofrecer soporte especial para ejecutar servicios diseminados en diferentes nodos para ser ejecutados en forma de transacción atómica
    - Composición de servicios: Web Services, mashups (ej. Google maps enhaced with extra information such as trip planners or wheather forecasts)
    - Confiabilidad: proveer funciones mejoradas para construir aplicaciones confiables (ej garantizar que cualquier mensaje enviado por un proceso tenga garantizado que el mensaje sea recibido por todos o por ninguno)

Objetivos de diseño

- Compartir recursos: acceder y compartir recursos remotos (periféricos, storage, archivos, servicios, etc). Por motivos económicos, para facilidad de colaboración e intercambio de información. Los ejemplos mas ilustrativos: BitTorrent (File-sharing peer-to-peer), Sharing Folders worldwide (permite ubicar archivos en carpetas compartidas gestionadas por terceros en algun lado de Internet).

Canonical examples

- Cloud-based shared storage and files
- Peer-to-peer assisted multimedia streaming
- Shared mail services (think of outsourced mail systems)
- Shared Web hosting (think of content distribution networks)

Objetivos de diseño

- Transparencia de la distribucion: esconder el hecho de que los procesos y recursos están fisicamente distribuidos a través de múltiples redes a lo largo del planeta, hacerlo lo mas invisible posible.

Types

Transparency	Description
Access	Hide differences in data representation and how an object is accessed
Location	Hide where an object is located
Relocation	Hide that an object may be moved to another location while in use
Migration	Hide that an object may move to another location
Replication	Hide that an object is replicated
Concurrency	Hide that an object may be shared by several independent users
Failure	Hide the failure and recovery of an object

Objetivos de diseño

Grado de transparencia: pretender una transparencia total puede ser demasiado.

- Hay latencia de comunicación que no se puede ocultar
- Esconder todas las falla de redes y nodos es (teórica y prácticamente) imposible
  - No se puede distinguir una computadora lenta de una que falla
  - No se puede estar seguro de que un server realmente hizo una operación antes de caerse
- Transparencia completa tendrá costo en la performance, exponiendo a su vez que el sistema es distribuido
  - Mantener réplicas exactamente actualizadas con el master, toma tiempo
  - Hacer las operaciones de escritura a disco para tolerancia a fallas tambien tiene costo alto

Exposing distribution may be good

- Making use of location-based services (finding your nearby friends)
- When dealing with users in different time zones
- When it makes it easier for a user to understand what's going on (when e.g., a server does not respond for a long time, report it as failing).

Objetivos de diseño

- Que el sistema sea abierto (Openess): un SD es un sistema que ofrece componentes que puedan ser fácilmente utilizados o integrados con otros sistemas abiertos. Que sea abierto significa que sus componentes se adhieren a reglas estándar que describen la sintaxis y semántica de lo que los componentes tienen para ofrecer. Un enfoque general es usar un IDL (interfaz definition language)

    - Los sistemas deben cumplir ***Interfases*** bien definidas
    - Deben tener ***Interoperabilidad*** (acoplar con otros sistemas de otros fabricantes)
    - Deben soportar ***Portabilidad*** de aplicaciones (ser capaz de correr en diferentes plataformas)
    - Deben ser fácilmente ***Expandibles*** (que se puedan agregar o modificar componentes sin tener que modificar aquellos que no se alteran)

Objetivos de diseño

- Escalabilidad: en la medida que se masificó Internet, fuimos siendo testigos de como las aplicaciones y servicios están siendo ubicados actualmente en la nube, con lo cual ya no se requiere tanto poder de procesamiento en las computadoras locales de la red, que ahora pueden ser smartphones o tablets.

  La escalabilidad de un sistema puede medirse a través de tres dimensiones:

  - De Tamaño (***size scalability***): es escalable con respecto a su tamaño, se pueden agregar mas usuario y recursos al sistema sin una pérdida notable de performance
  - Geográfica (***geographical scalability***): es aquel en el cual donde los usuarios y los recursos pueden estar muy alejados pero el delay inevitable de comunicación apenas se nota
  - Administrativa (***administrative scalability***): es aquel que puede ser fácilmente administrado incluso si abarca muchas organizaciones administrativas independientes

Objetivos de diseño

Trampas : hay muchos temas a considerar en el diseño de los SD lo cual hace que pareczca que solo se va a lograr mucha complejidad. Sin embargo, siguiendo una serie de principios de diseño se pueden desarrollar SD que fuertemente cumplan estos objetivos trazados.

Los SD difieren de los sistemas tradicionales dado que sus componentes están dispersos a través de la red, y eso debe ser tenido en cuenta.

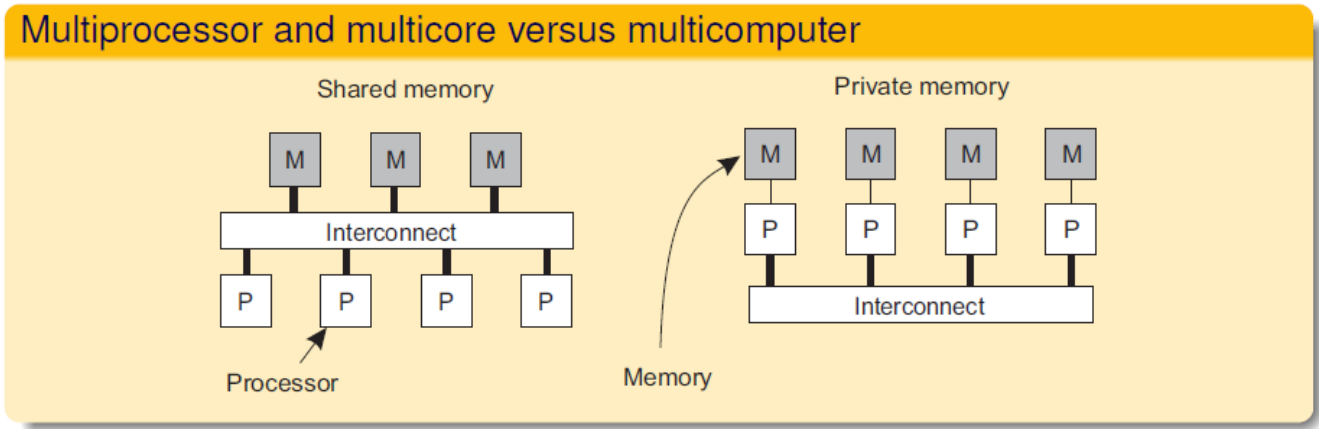
False (and often hidden) assumptions

- The network is reliable
- The network is secure
- The network is homogeneous
- The topology does not change
- Latency is zero
- Bandwidth is infinite
- Transport cost is zero
- There is one administrator

Todas relacionadas con propiedades que son propias de los SD: confiabilidad, seguridad, heterogeneidad, topología de la red, latencia y ancho de banda, costos de transporte y dominios de administracion

Tipos de SD

- 1 - Computación distribuida de alta Performance
  - Comenzó con la ***computación paralela***
  - Primero con máquinas multiprocesador: memoria compartida, uso de semáforos, son difíciles de escalar en numero de cores
  - Luego con sistemas multicomputador: pasaje de mensajes por ausencia de memoria compartida (más dificiles de programar)
  - Comenzaron de esta forma a hace pasaje de mensajes para alterar la memoria

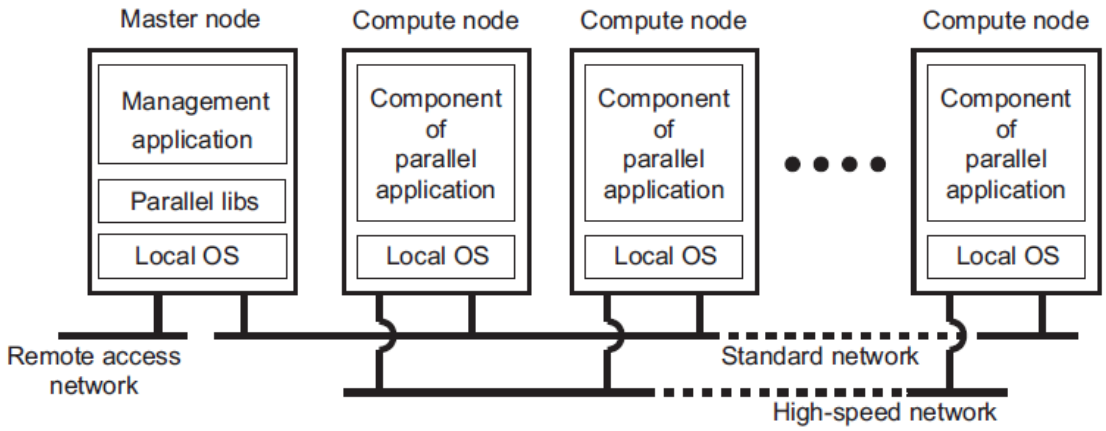


Tipos de SD

- Computación distribuida de alta Performance

  Cluster computing

  - Hardware homogéneo: mismo OS, hardware casi idéntico
  - Nodo master único que maneja el sistema distribuido
  - Computadoras unidas por una red standard de alta velocidad
  - En casi todos los casos se utiliza para programación paralela donde un programa unico de cómputo intensivo se corre en paralelo en múltiples máquinas.
  - Ej: Linux Boewulf clusters, MOSIX (enfoque simétrico, provee alta transparencia)

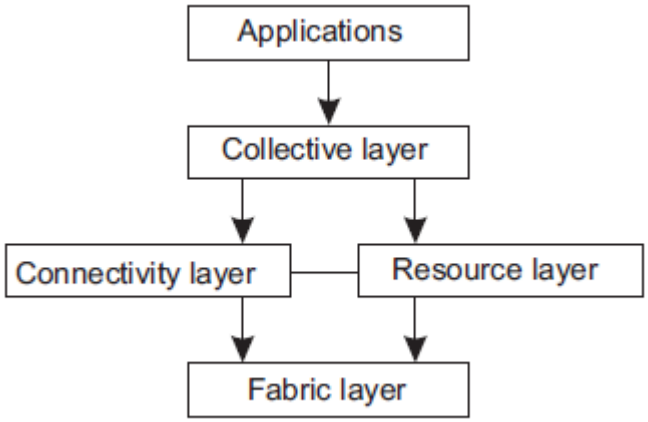


Tipos de SD

- Computación distribuida de alta Performance

  Grid computing

  - Nodos heterogéneos en cuanto a hardware, OS, red y dominio de administración
  - Dispersos a través de diferentes organizaciones
  - Pueden abarcar facilmente una red amplia
  - Usan concepto de *organizaciones virtuales*: agrupamiento de procesos que son autorizados la ubicación de recursos y de usuarios autorizados para su uso
  - Los recursos pueden ser servidores (tal vez en cluster), storage, DBs, sensores, etc



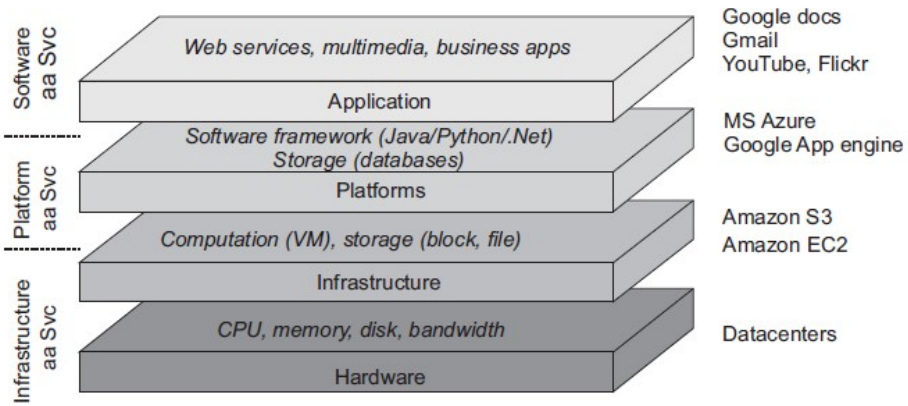
Application: Contains actual grid applications in a single organization and wich make use of the grid computing environment.
Collective: Handles access to multiple resources: discovery, scheduling, replication.
Resource: Manages a single resource, such as creating processes or reading data. It uses functions provided by connectivity layer and call directly to interfaces on fabric layer
Connectivity: Communication/transaction protocols, e.g., for moving data between resources. Also various authentication protocols. Delegating rights from a user to programs is an important function supported in connectivity layer.
Fabric: Provides interfaces to local resources (for querying state and capabilities, locking, etc.)

Tipos de SD

- Computación distribuida de alta Performance

  Cloud computing

  - Son pooles de recursos virtualizados de fácil uso y accesibilidad , provistos normalmente por los data center.
  - Cuales recursos y como son utilizados, se configura en forma dinámica, proveyendo las bases para escalabilidad: si se requiere mas poder de cómputos el usuario puede adquirir mas recursos. De esta forma cloud computing se basa en pay-per-use model garantizándose el servicio mediante *service level agreements* (SLAs)



Application: Actual applications, such as office suites (text processors, spreadsheet applications, presentation applications). Comparable to the suite of apps shipped with OSes..
Platform: Provides higher-level abstractions for storage and such. Example: Amazon S3 storage system offers an API for (locally created) files to be organized and stored in so-called buckets.
Infrastructure: Deploys virtualization techniques. Evolves around allocating and managing virtual storage devices and virtual servers.
Hardware: Processors, routers, power and cooling systems. Customers normally never get to see these.

Tipos de SD

- Computación distribuida de alta Performance

  Cloud computing

  - Los proveedores de Cloud computing ofrecen estas capas a sus clientes a través de varias interfases (de linea de comando, APIs, interfases WEB, etc) apuntando a tres tipos de servicios
    - **Infrastructure-as-a-Service** (**IaaS**) abarcando las capas de hardware e infraestructura
    - **Platform-as-a-Service** (**PaaS**) abarcando la capa de plataforma
    - **Software-as-a-Service** (**SaaS**) donde se accede a las aplicaciones

  - La computación en la nube ya no es una utopía, ciertamente ya es una alternativa seria para mantener grandes infraestructuras locales, tercerizando servicios. Sin embargo, todavía hay mucho margen de mejora (obstáculos a la hora de depender de un proveedor, de la disponibilidad de los servicios, de cuestiones de privacidad de datos, seguridad, etc)

Tipos de SD

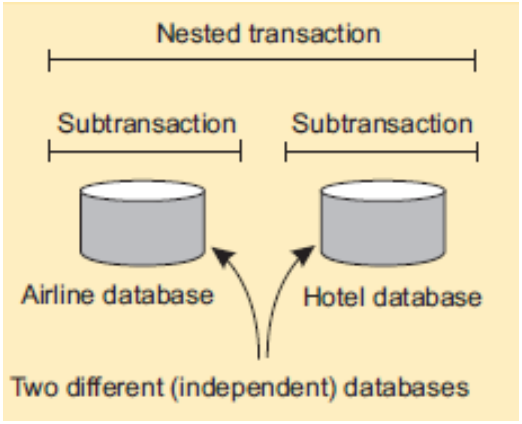
- 2 - Sistemas de información distribuida
    - Las organizaciones usaban las aplicaciones en red asiduamente, pero lograr la interoperabilidad de las mismas era una tarea dificultosa.
    - Muchas de las soluciones de middleware actuales se deben a esfuerzos para hacer mas fácil integrar aplicaciones en el ámbito de una empresa y luego se generalizaron
    - Las aplicaciones en red son aquellas que corren en un servidor haciendo sus servicios disponibles a clientes.
    - En una *integración simple* los clientes combinan requerimientos a múltiples aplicaciones en red , los envian, recolectan el resultado, y lo presentan en forma coherente al usuario.
    - El siguiente paso → permitir comunicación de las aplicaciones entre ellas llevandonos a una *Enterprise Application Integration* (EAI)

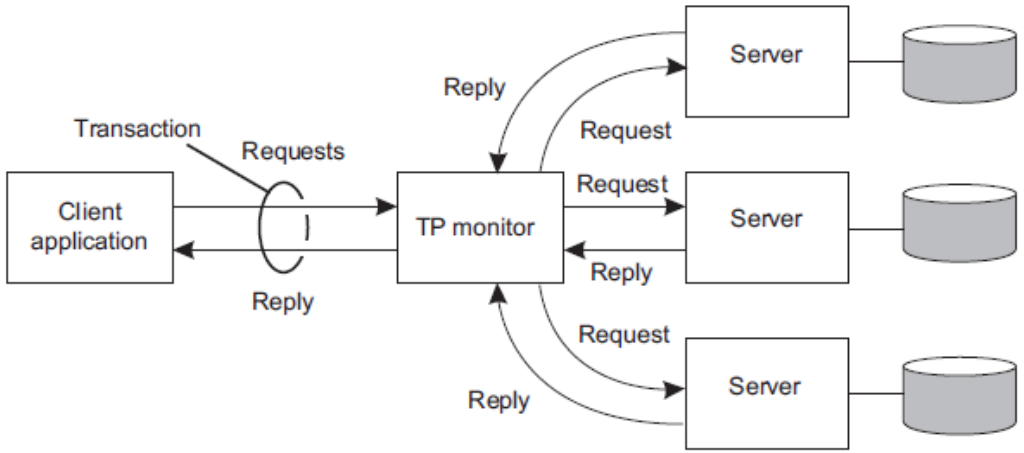
Tipos de SD

- Sistemas de información distribuida

  Ejemplo EAI – TPM (Transaction Processing Monitor)

  - En muchos casos los datos involucrados en una transacción se distribuyen a través de varios servidores. El TP Monitor es el responsable de coordinar la ejecución de la transacción.
  - Ejemplo: una transacción para planear un viaje que involucra la reserva de tres vuelos, puede ser dividida en tres subtransacciones. Cada una manejada en form independiente.
  - Una aplicación que quiere coordinar varias subtransacciones hace uso del TPM





Tipos de SD

- Sistemas de información distribuida

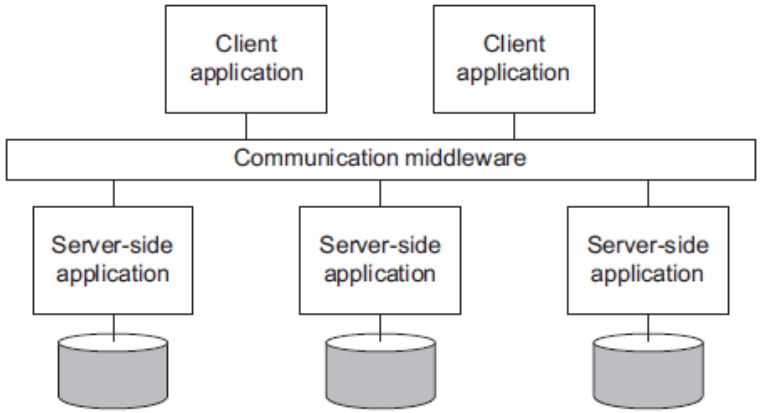
EAI y Middleware – Modelos de comunicación

Requieren que el llamador y el que es llamado estén bloqueados a la espera

- Remote Procedure Call (RPC): Los requerimientos son enviados como una llamada a procedimiento local, encapsulados en un mensaje, procesados y la respuesta es encapsulada en un mensaje y devuelta como resultado de la llamada.
- Invocación a métodos remótos (RMI): Similar a RPC excepto porque opera con objetos en lugar de funciones.

Permite desacoplar en tiempo y espacio

- Message Oriented Middleware (MOM): Los mensajes son enviados por las aplicaciones (publishers) a puntos de contacto lógicos, normalmente descritos como asunto. Del mismo modo las aplicaciones pueden mostrar interés en un determinado tipo de mensaje o asunto (subscribers). Publisher/subscribe systems



Tipos de SD

- 3- Sistemas pervasivos (ominpresentes)
    - Emergen de la nueva generación de sistemas distribuidos donde los nodos son pequeños, móviles y usualmente integrados en un sistema mas grande
    - Caracterizados por el hecho de que el sistema se mezcla naturalmente con el entorno de usuario.
    - Lo que los distingue de los sistemas de computacion y de información es que es mucho mas difusa la separación entre usuarios y componentes del sistema, dado que están equipados con *sensores* que toman varios aspectos del comportamiento del usuario, y también pueden tener *actuadores* para devolver comportamiento.
    - Se pueden distinguir 3 subtipos (aunque hay solapamiento entre ellos)
        - ***Computación ubicua:*** *pervasive*, continuamente presente, hay una interaccion continua entre sistema y usuario
        - ***Mobile :*** *pervasive* pero con énfasis en el hecho de que los dispositivos son inherentemente móviles
        - ***Sensor networks:*** *pervasive*, pero con énfasis en el detección y actuación reales del entorno.

Tipos de SD

- Sistemas pervasivos

  Computación ubicua

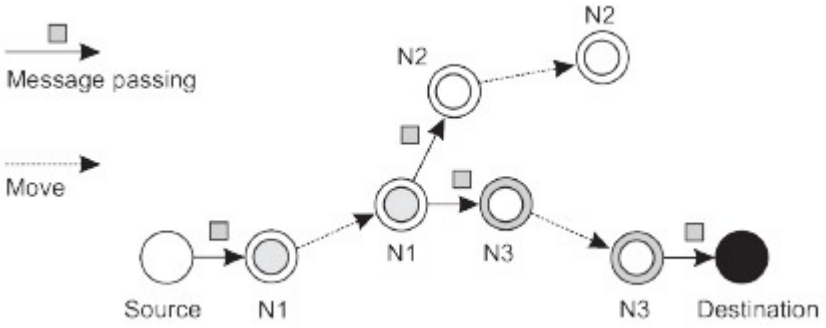
  - Requerimientos principales
    - ***Distribución:*** dispositivos en red, distribuidos, accesibles en forma transparente
    - ***Interacción:*** la interaccion entre usuarios y dispositivos es discreta, no obstructiva. El debe ignorar que su comportamiento produce entradas a un sistema de computadora. Computación ubicua *oculta* las interfases
    - ***Conocimiento del contexto:*** el sistema está en conocimiento del contexto del usuario, para optimizar la interacción. El sistema deberá poder sensar el qué, quién, cómo y dónde (contexto) Raw data → nivel de abstraccion
    - ***Autonomia:*** Sistema opera en forma autonoma y sin intervención humana y los dispositivos se auto gestionan. Ej: DHCP, PnP, auto-update
    - ***Inteligencia:*** El sistema como un todo puede manejar un amplio rango de acciones e interacciones. Ej: Entrada incompleta, cambios de entorno, etc

Tipos de SD

- Sistemas pervasivos

  Computación móvil (mobile computing)

  - Realizada mediante una variedad de dispositivos: smartphones, tablets, equipamiento en autos, gps, etc). Todos con característica wireless
  - Móvil implica que se espera que la ubicación del dispositivo varíe a lo largo del tiempo. Esto implica cambio de servicios locales, y publicacion de la ubicación para poder ser alcanzado. La clave: descubrimiento (*discovery)* Dónde está un servicio y Quien ofrece servicios.
  - Son redes tolerantes a la disrupción (*disruption-tolerant networks*) en las cuales la conectividad entre dos nodos no puede ser garantizada. Llevar un mensaje desde un nodo a otro puede ser problemático, ausencia de rutas estables.



Disruption-Tolerant networks

- Inundación de paquetes para, eventualmente, alcanzar el destino
- Un nodo intermedio almacena un mensaje recibido, hasta que encuentra otro nodo al que se lo pueda pasar

Tipos de SD

- Sistemas pervasivos

Computación móvil (mobile computing) - Patrones de movilidad

- El análisis del comportamiento social nos lleva a darnos cuenta de que la computación móvil está altamente acoplada al comportamiento humano.
- Con el advenimiento de los celulares, los expertos buscan combinar su uso con el comportamiento y el diseminamiento de la información en las llamadas ***pocket-switched networks***.
- Son redes donde los nodos son formados por los dispositivos de la gente y los enlaces se forman cuando dos personas se encuentran, permitiendo a sus dispositivos intercambiar datos.
- Estructura de un grupo social: dos personas son **amigos** (interactuan frecuentemente) o **extraños** (baja interactividad)
- Para asegurarse que Alice le pueda entregar un mensaje a Bob:
  - Alice le entrega el mensaje a cada uno de sus amigos
  - Los amigos le pasan el mensaje a sus amigos y asi sucesivamente hasta que el mensaje llega a Bob
  - No le pasa el mensaje a sus extraños porque no hace mayor diferencia.

Tipos de SD

- Sistemas pervasivos

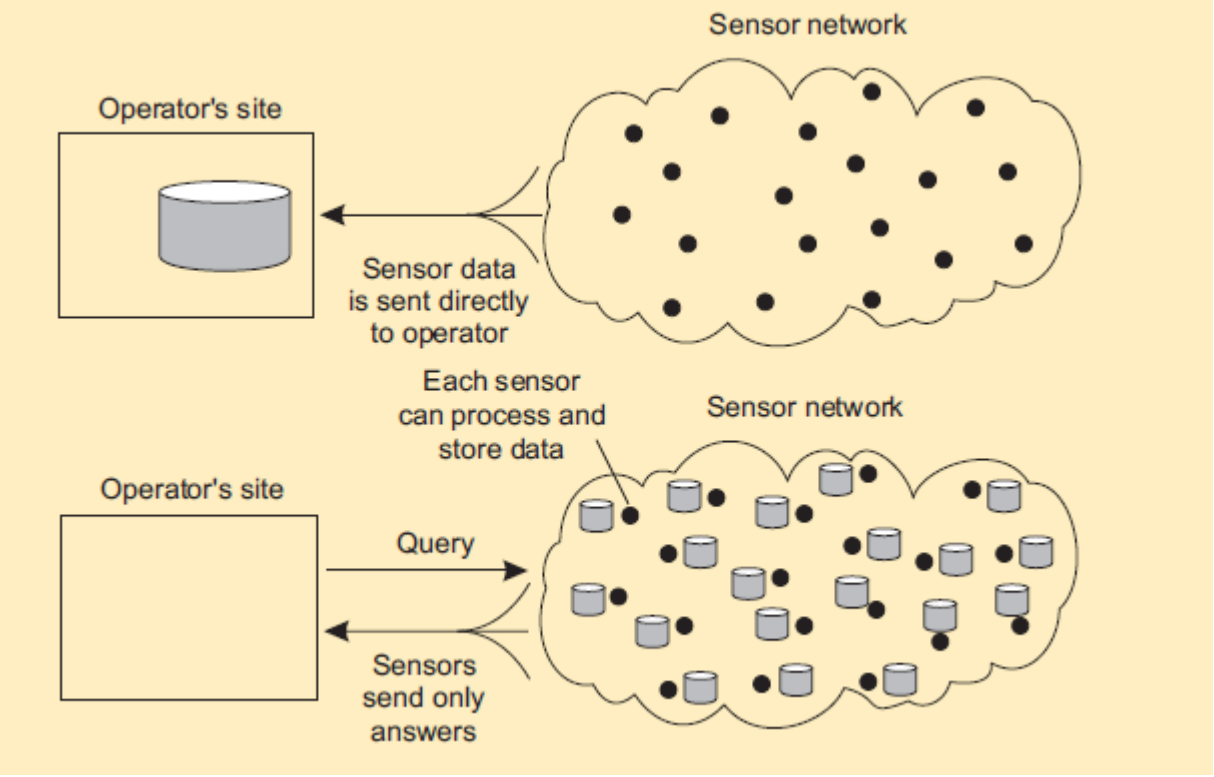
  Redes de sensores (sensor networks)

  - Consiste generalmente de cientos o miles de nodos relativamente pequeños cada uno equipado con uno o mas ***sensores***.
  - Adicionalmente los sensores pueden actuar como ***actuadores*** (ej. Sensores de incendio que activan la lluvia cuando detectan fuego)
  - Hacen uso de la comunicación wireless y generalmente son alimentados por batería. Esto hace que su diseño deba ser eficiente por los escasos recursos
  - Son mas que sólo una conexión de dispositivos de entrada. Por lo general los nodos colaboran para procesar los datos relevados en una forma específica.
  - Cada nodo contiene una capa de software encima del HW que permite : acceso de bajo nivel a la red, acceso a los sensores y actuadores, manejo de memoria y ocasionalemente: acceso a la ubicación (gps), almacenamiento, comunicación compleja.
  - Pueden permitir direccionar la dirección física de un nodo, de un vecindario (o grupo de nodos) o del sistema completo

Tipos de SD

- Sistemas pervasivos

  Redes de sensores (sensor networks)



Resumen

Sistemas distribuidos consisten de multiples computadoras autónomas trabajando juntas para dar la aparencia de un sistema único y coherente

Este comportamiento colectivo y coherente se logra mediante el establecimiento de protocolos independientes de la aplicación, en lo que es conocido como ***middleware***: una capa de software situada entre el sistema operativo y las aplicaciones distribuidas. Los protocolos la incluyen para comunicación, transacciones y confiabilidad.

Los objetivos de diseño se agrupan en : ***compartir recursos***, brindar ***transparencia de distribución***, hacer el ***sistema abierto***, y la ***escalabilidad***.

La transparencia tiene un costo y además en algunos casos no es deseable que sea total, con lo cual se habla de un ***grado de transparencia***. La transparencia y la escalabidad por lo general son inversas.

Otro aspecto del diseño es tener en cuenta las ***trampas*** que puede conllevar creer que la red es fiable, o que no tiene latencia, o que es estática, segura, homogenea.

Resumen

Tipos de sistemas distribuidos

- Sistemas de ***computación distribuida*** (de alta performance)
  - ***Parallel computing*** (computación paralela)
  - ***Cluster computing***
  - ***Grid computing***
  - ***Cloud computing***

- Sistemas de ***información distribuida***

- Sistemas ***pervasivos***
  - ***Computación ubicua***
  - ***Computación móvil***
  - ***Redes de sensores***