introduction

distributed system definition

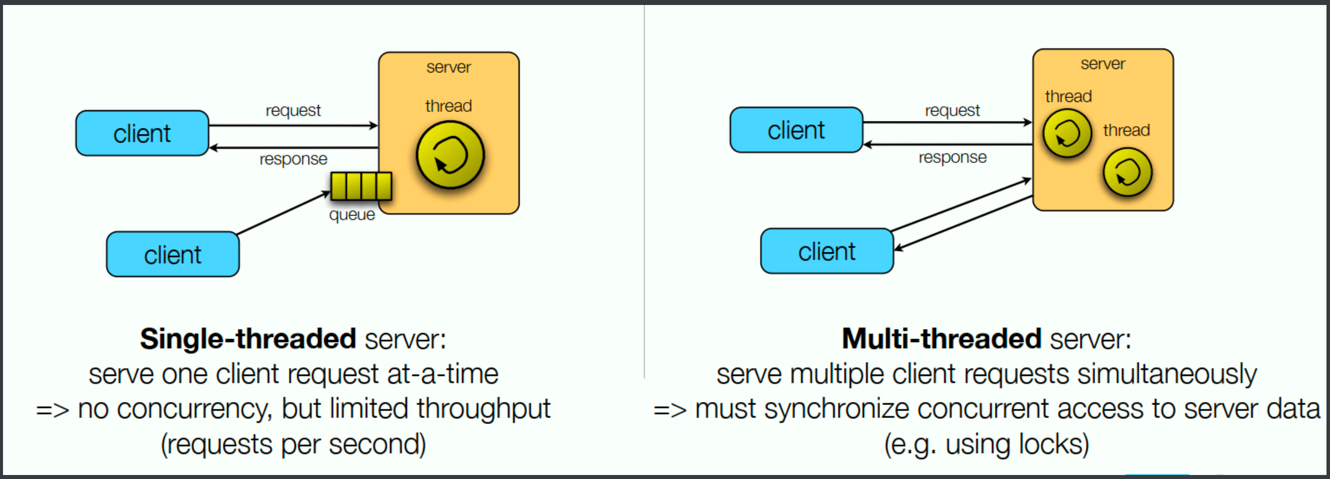
- sw/hw as components
- interconnected
- message passing (communication)

challenge: concurrency, global notion of time, fault tolerance etc.

design goals

1. resource sharing

   implies concurrency, client-server paradigm:

   Single-threaded server:
   serve one client request at-a-time
   => no concurrency, but limited throughput
   (requests per second)

   Multi-threaded server:
   serve multiple client requests simultaneously
   => must synchronize concurrent access to server data
   (e.g. using locks)

2. transparency

   motivation: heterogeneity

   middleware provides a uniform high-level API

   (make features unobservable for the user, "don't know...")

form of transparency	**meaning**	**example**
access ...	local & remote resources are accessed in the same way	dropbox file integration
location ...	access resources without knowing their physical, network location	domain name / URLs
relocation ...	move resources without affecting ongoing operations	phone roam
replication ...	don't have to know whether a resource is replicated	
concurrency ...	avoid interference from concurrent access	transaction
failure ...	hide the fact that some parts have failed	

degree of transparancy? :

- full transparency is not always possible (e.g. speed of light)
- understanding limitations might be what the users want

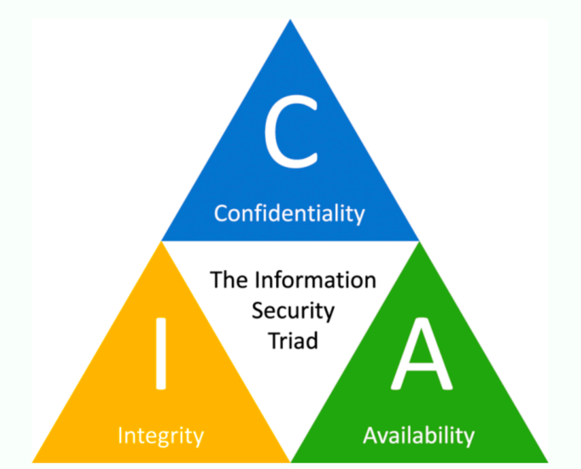
3. openness

   interoperable 互操作性, portable 可移植性, extensible 可扩展性:

   solutions:

   well-defined components, IDL, standardized protocols

4. dependability: fault tolerance
5. security



(Image source: Nikander, Jussi & Manninen, Onni & Laajalahti, Mikko. (2020). Requirements for cybersecurity in agricultural communication networks. Computers and Electronics in Agriculture.)

   authentication, authorization

6. scalability

   hide latencies, load balancing, caching

types of distributed systems

1. high performance computing

   parallel / concurrency

   parallel / distributed hardware

2. distributed information system

   2.1 transaction processing

   ACID properties of transactions:

   Atomic: indivisible to the outside world

   consistent: does not violate application invariants

   isolated: when concurrent, do not interfere with each other

   durable: commits are permanent

   2.2 enterprise application integration

   service-oriented architectures

challenges in distributed systems

- Common false assumptions made by first time distributed systems developers:
  - 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**.

L. Peter Deutsch
(ACM Fellow, Sun Microsystems Fellow)