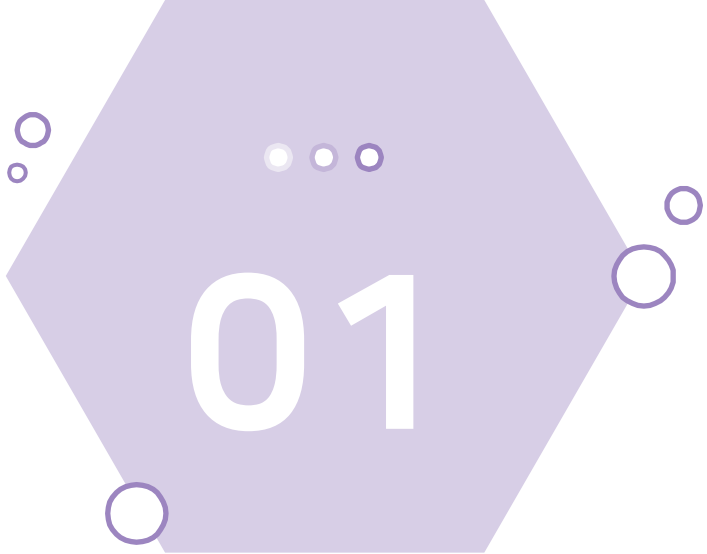
13강. 분산 운영체제

방송대 컴퓨터과학과
김진욱 교수

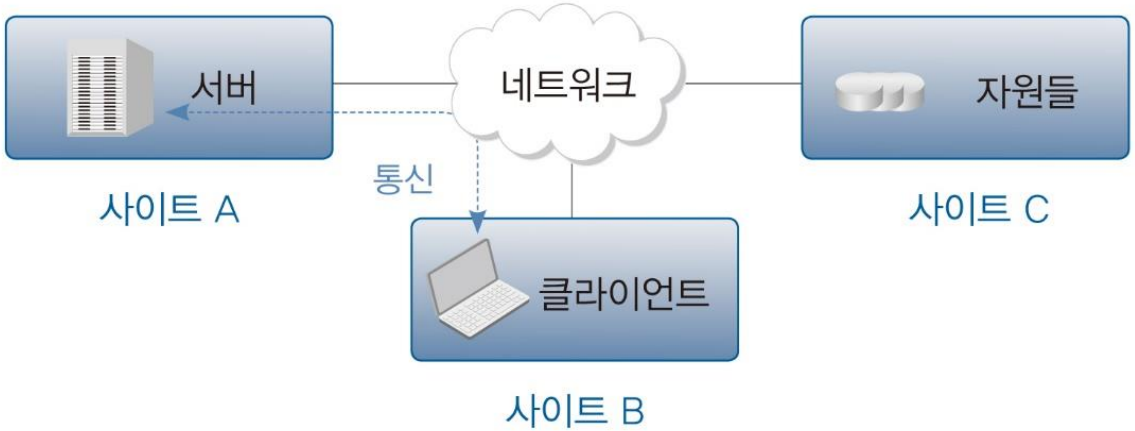
목차



분산 운영체제의 개요

분산 시스템

- 분산 시스템
  - 메모리나 클럭을 물리적으로 공유하지 않는 프로세서들의 집합
  - 네트워크로 연결되어 상호 협력 가능



프로세서

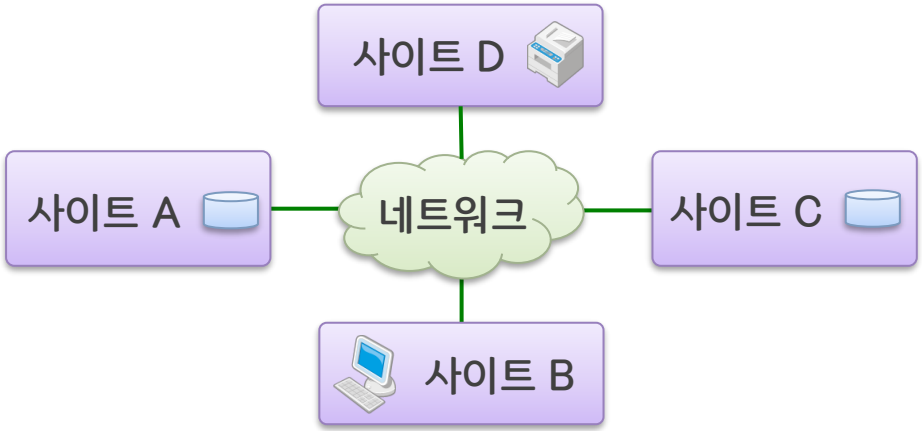
- 범용 메인프레임, 중형 컴퓨터, 소형 컴퓨터 등을 통칭
- 사이트, 호스트, 노드 등으로도 불림

분산 시스템의 목적

- 자원 공유
- 연산속도 향상
- 신뢰성 향상
- 통신의 용이성

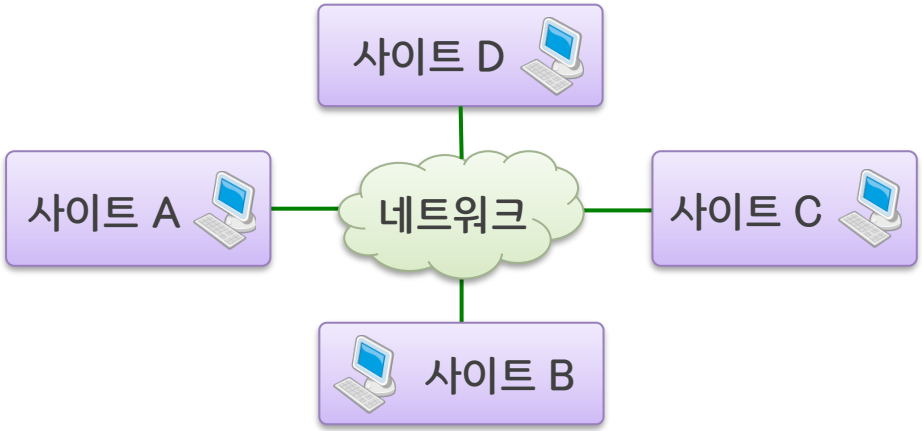
분산 시스템의 목적

- 자원 공유
  - 연결된 다른 사이트의 자원을 사용
  - 예: 원격 사이트의 파일 공유, 원격 사이트에서의 프린팅, 분산 DB의 정보 처리 등



분산 시스템의 목적

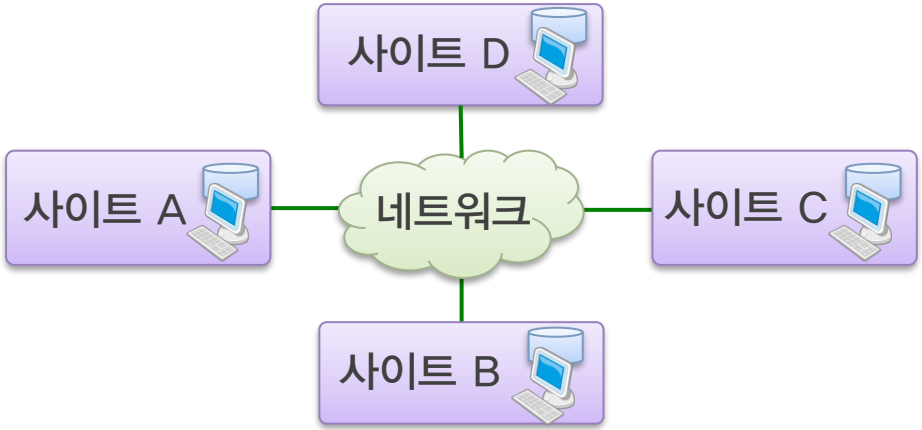
- 연산속도 향상
  - 분할이 가능한 작업을 분산 시스템의 여러 사이트에 분산시켜 동시 처리
  - 부하 공유를 통해 과부하 해소 및 전체 처리속도 향상



분산 시스템의 목적

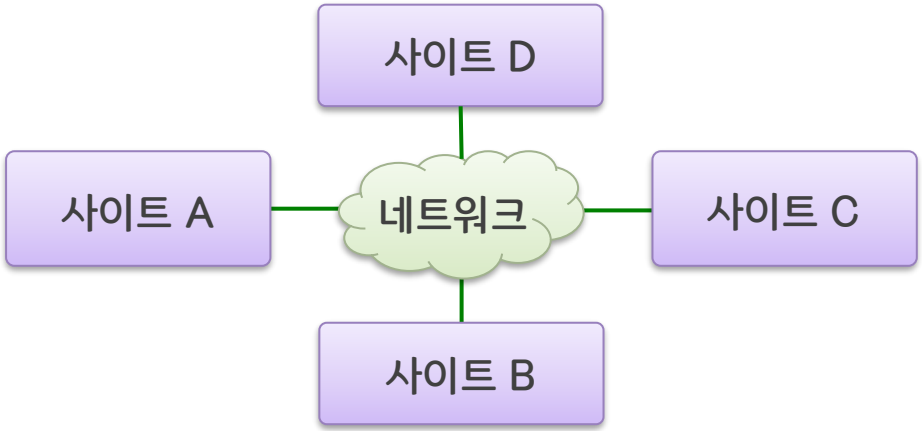
신뢰성 향상

- 일부 사이트에서 장애가 발생하더라도 전체 시스템의 동작이 멈추지 않음
- 장애 검출, 장애 시스템의 기능을 다른 시스템으로 이동,
  복구 후 자연스런 복귀 등 필요
- 하드웨어나 데이터의 중복을 통한 해결 가능



분산 시스템의 목적

- 통신의 용이성
  - 통신 네트워크로 연결된 사이트들의 사용자간 정보 교환 가능
  - 하위 수준: 시스템간 메시지 전달
  - 상위 수준: 파일 전송, 로그인, 메일 전송, 원격 프로시저 호출(RPC) 등



분산 시스템의 네트워크 구성

네트워크 구성

- 완전 연결 네트워크, 부분 연결 네트워크
- 비교 기준: 구축비용, 통신비용, 가용성

구축비용

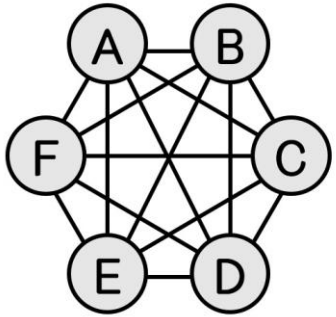
사이트들을 물리적으로 연결하는 비용

통신비용

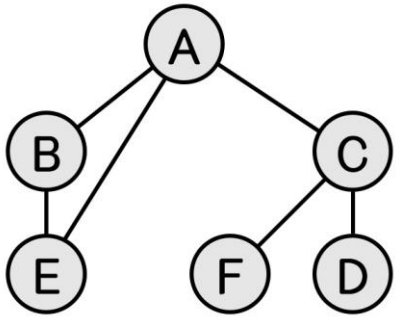
메시지를 보내는 데 쓰이는 시간과 비용

가용성

링크나 사이트의 고장 시 접근 가능성



완전 연결 네트워크



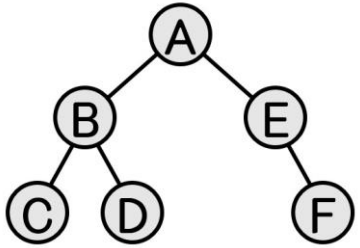
부분 연결 네트워크

기준 \ 구성	완전 연결 네트워크	부분 연결 네트워크
구축비용	많음	적음
통신비용	적음	많음
가용성	높음	낮음

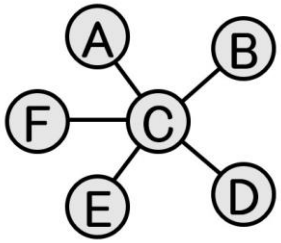
분산 시스템의 네트워크 구성

네트워크 구성

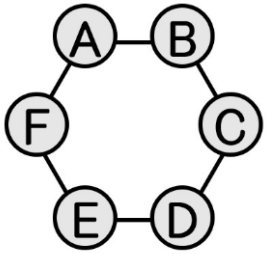
- 부분 연결 네트워크



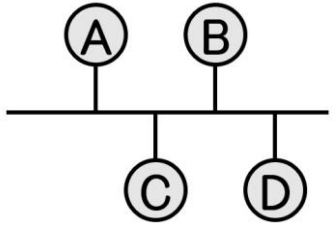
트리 구조 네트워크



스타형 네트워크



링형 네트워크



버스형 네트워크

	트리 구조 네트워크	스타형 네트워크	링형 네트워크	버스형 네트워크
통신비용:	비교적 저렴	저렴	높음	저렴
가용성 :	낮음	높음 (단, 중앙 사이트 제외)	비교적 높음	높음 (단, 버스 제외)

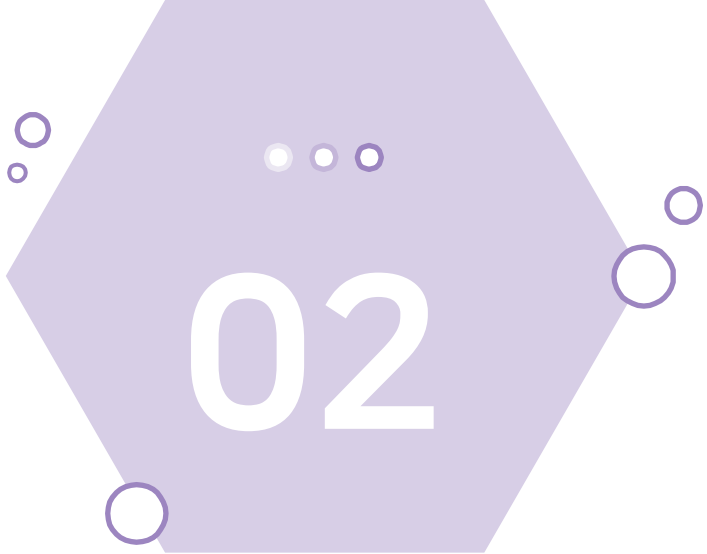
분산 운영체제

- 투명성(transparency) 제공
  - 사용자는 로컬 자원을 사용하는 것과 동일한 방식으로 원격지 자원을 사용
  - 분산 운영체제가 알아서 원격지 자원을 쓸 수 있도록 해줌

- 운영체제가 제공하는 기능
  - 데이터 이주
  - 계산 이주
  - 프로세스 이주

분산 운영체제

- 데이터 이주
  - 원격지의 데이터를 필요한 곳으로 전송하여 사용
  - 전체 전송 방식, 일부 전송 방식
- 계산 이주
  - 원격지의 대량의 데이터가 필요한 경우 원격지에서 처리(계산) 후 결과를 받음
  - 원격 프로시저 호출(RPC) 이용
- 프로세스 이주
  - 프로세스 자체를 원격지로 이주시킴으로써 부하분산, 계산속도 향상 가능



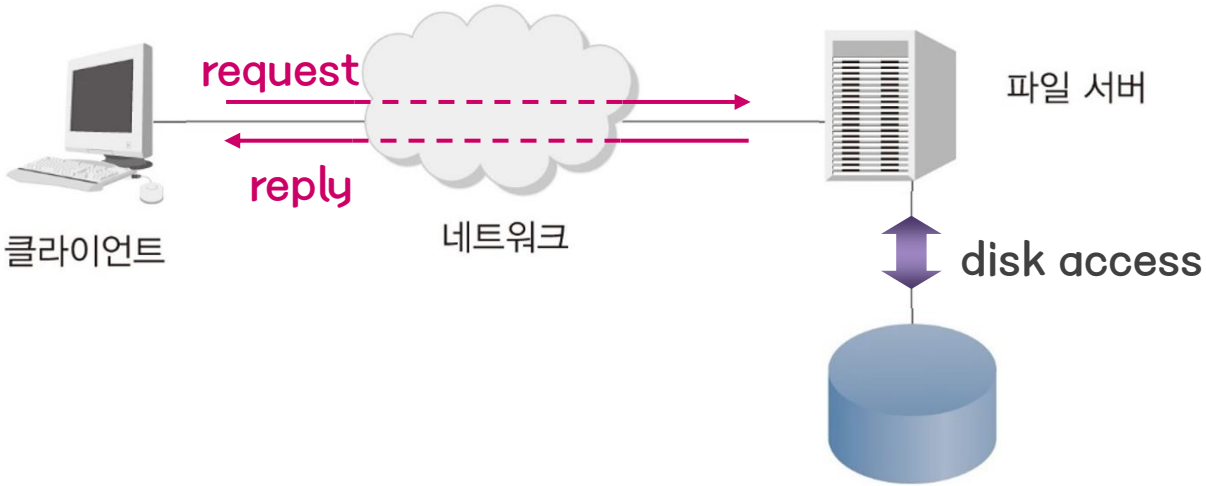
분산 파일 시스템

분산 파일 시스템

- 분산 파일 시스템(Distributed File System, DFS)
  - 클라이언트가 서버에 저장된 파일을 마치 로컬 파일인 것처럼 처리할 수 있는 파일 시스템
  - 투명한 DFS
    - » DFS의 클라이언트 인터페이스는 로컬 파일과 원격 파일을 구별하지 않음
- DFS의 네이밍(naming) 방식
  - 호스트 이름과 로컬 이름을 조합하는 방식
    - » host:local_name
  - 원격 디렉토리들을 로컬 디렉토리에 붙이는 방식
    - » 일관된 디렉토리 트리처럼 보임

분산 파일 시스템

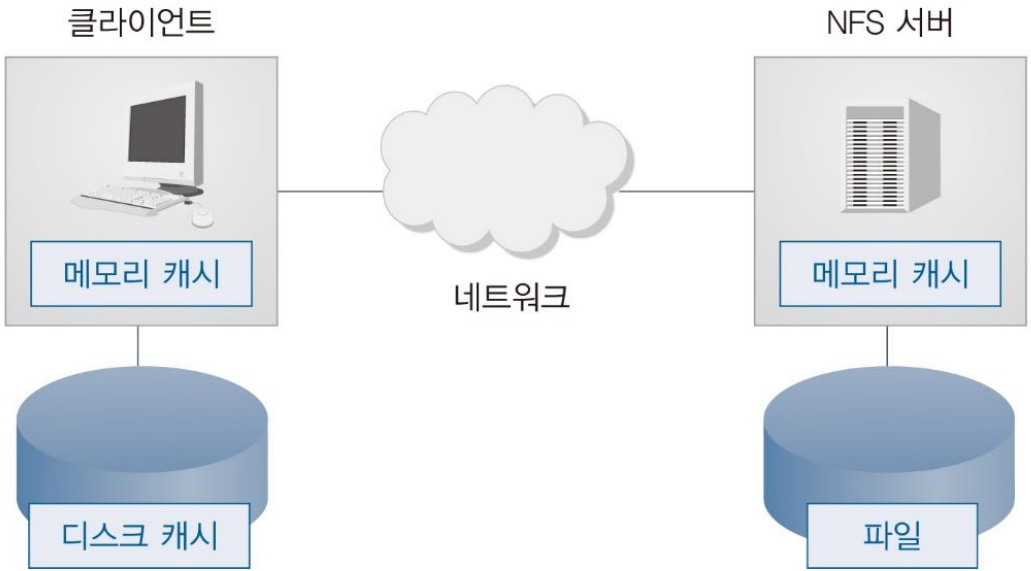
- 원격 파일에 대한 접근 요청
  - 원격 서비스 메커니즘 : RPC를 통해 구현
  - 캐시 활용 방법

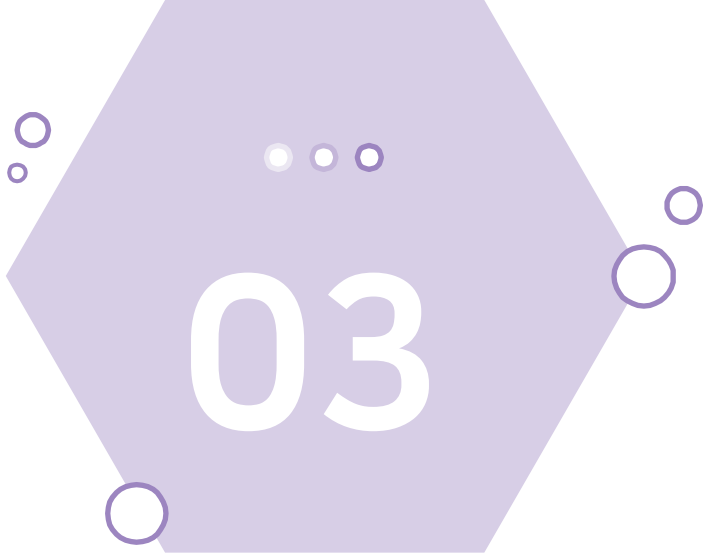


분산 파일 시스템

원격 파일에 대한 접근 요청

- 원격 서비스 메커니즘
- 캐시 활용 방법 : 캐시 교체 정책, 캐시 업데이트 정책, 캐시 일관성 문제 등 고려





분산 메모리

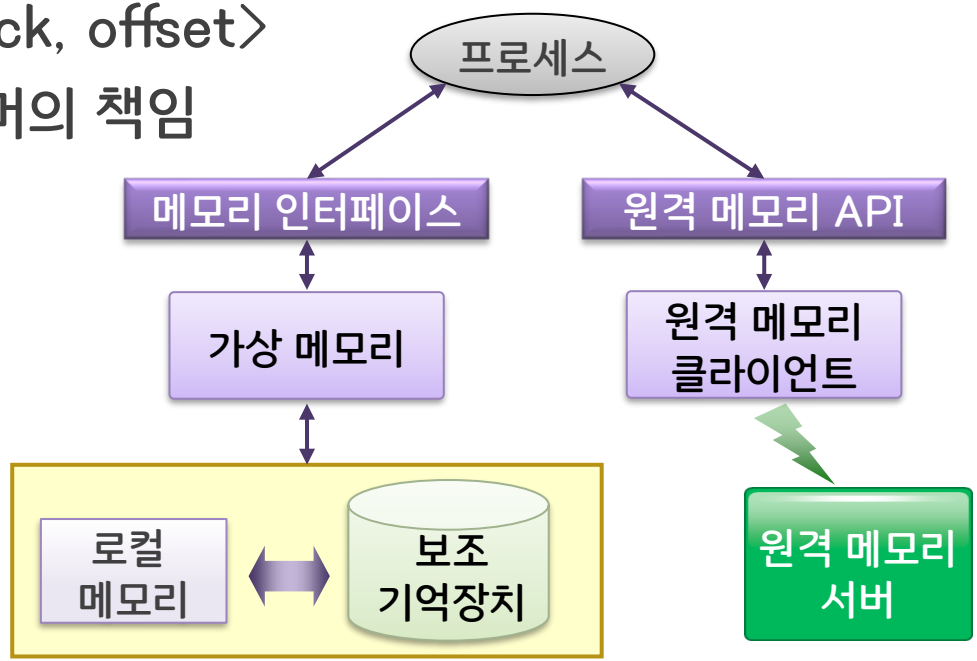
분산 메모리

- 분산 메모리
  - 메모리 인터페이스가 원격 컴퓨터의 메모리를 참조하도록 하는 것
  - 두 가지 모델: 원격 메모리, 분산 공유 메모리

분산 메모리

원격 메모리

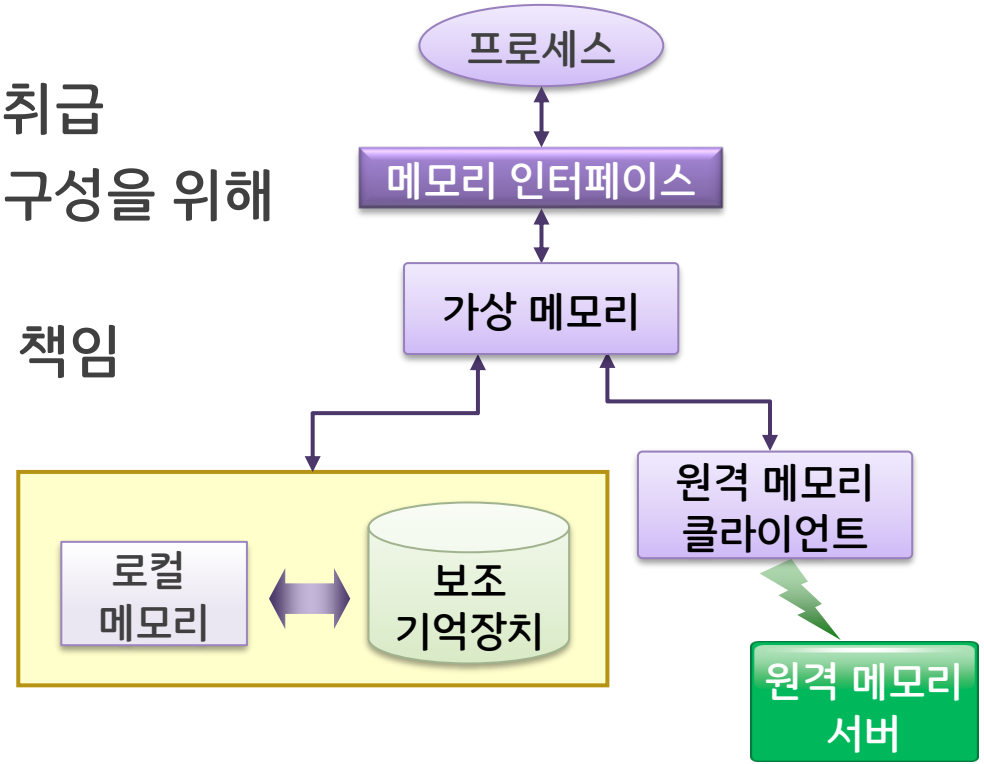
- 일반적인 메모리 인터페이스와 다른 원격 메모리 API 사용
- <(net#, host#, port#), block, offset>
- 데이터 일관성 유지: 프로그래머의 책임



분산 메모리

원격 분산 공유 메모리

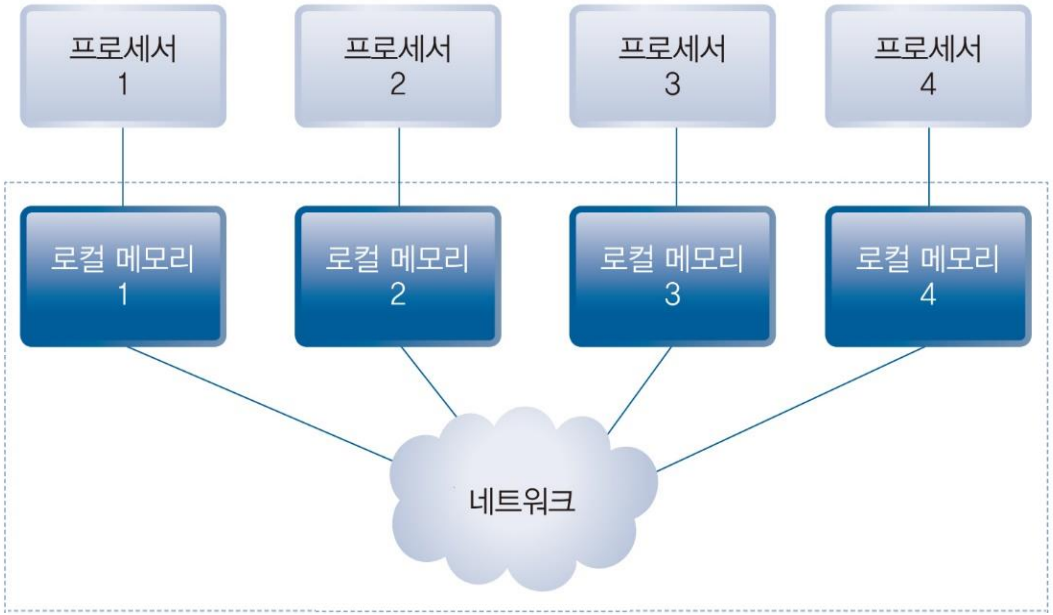
- 가상 메모리 인터페이스를 이용
- 원격 메모리를 로컬 메모리처럼 취급
- 메모리 관리자는 가상주소 공간 구성을 위해 네트워크 네이밍과 투명성 고려
- 데이터 일관성 유지: 운영체제의 책임



분산 메모리

원격 분산 공유 메모리의 예

- NUMA (Non-Uniform Memory Access) 구조
  - » 각 프로세서가 로컬 메모리를 가짐
  - » 데이터 저장 위치가 로컬 메모리냐 원격 메모리냐에 따라 접근 속도에 차이가 발생

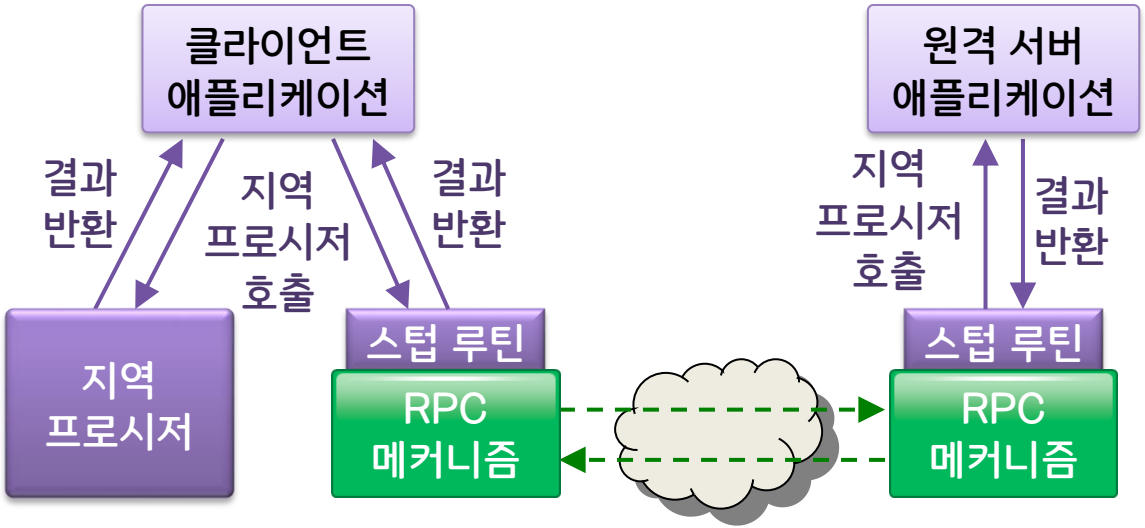


04

원격 프로시저 호출

원격 프로시저 호출(Remote Procedure Call, RPC)

- 한 컴퓨터에서 작동하고 있는 애플리케이션이 다른 컴퓨터에 있는 프로시저를 호출할 수 있도록 하는 클라이언트/서버 메커니즘



스텁(stub) 루틴

파라미터나 결과를
메시지로 만든 후
네트워크를 통해 전달

원격 프로시저 호출의 구현

theClient

```
int main(…) {
      ……
      localF(…);
      remoteF(…);
      ……
}
```

```
localF(…) {
      ……
}
```

rpcServer

```
remoteF(…) {
      ……
}
```

원격 프로시저 호출의 구현

theClient

```
int main(…) {
     ……
     localF(…);
     remoteF(…);
     ……
}
```

```
localF(…) {
     ……
}
```

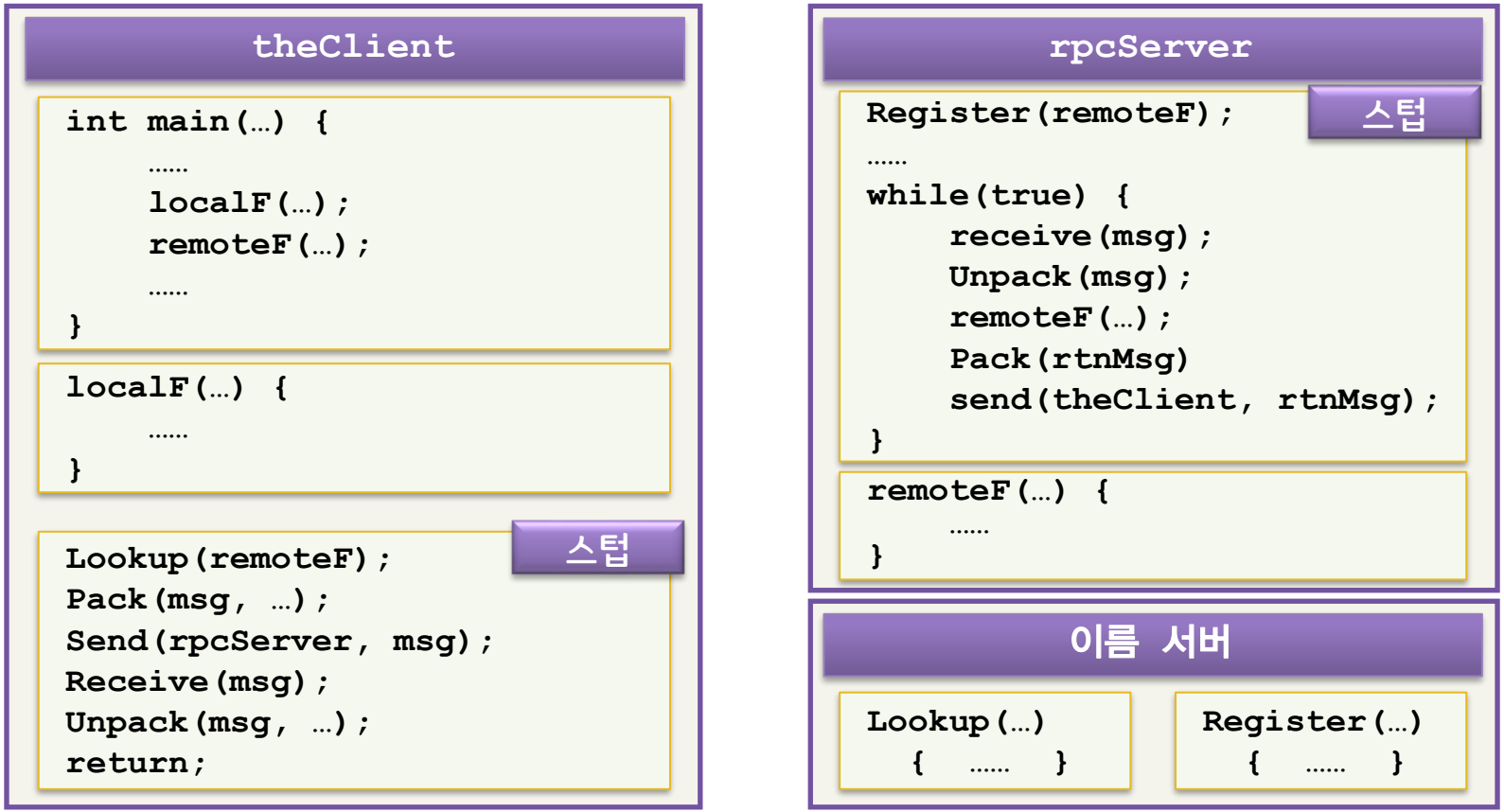
rpcServer

스텁

```
Register(remoteF);
……
while(true) {
     receive(msg);
     Unpack(msg);
     remoteF(…);
     Pack(rtnMsg)
     send(theClient, rtnMsg);
}
```

```
remoteF(…) {
     ……
}
```

원격 프로시저 호출의 구현



강의를 마쳤습니다.

다음시간에는

14강. 운영체제 보안