11 상위 계층의 이해

가천대학교 - 2019학년도 1학기 -

Contents

학습목표

- 세션 계층의 필요성과 세션 연결의 개념을 이해한다.
- 네트워크에서 데이터 표현의 일관성이 무엇인지 이해한다.
- 데이터를 압축하는 원리를 이해한다.
- 응용 환경에서의 클라이언트-서버 모델을 이해한다.

내용

- 상위 계층 소개
- 세션 계층
- 표현 계층
- 응용 계층

01_상위 계층 소개

❖ TCP/IP 계층 구조

- 5, 6, 7계층을 묶음으로 응용 프로그램으로 만들어 필요한 응용 기능 구현
- 예) 텔넷, FTP, 전자 메일 등 네트워크 응용 프로그램

- 계층 4까지 기능은 운영체제 내부에서 구현

그림 11-1 TCP/IP 계층 구조

02_세션 계층

세션 계층

- 응용 환경에서 전송 계층이 제공하는 서비스를 손쉽게 이용하기 위해 사용자의 논리적인 관점을 고려하여 단순한 사용자 인터페이스 제공 필요
- 기능이 제한적이라 다른 계층보다 상대적으로 간단한 계층

세션 계층의 기능

- 세션 연결의 설정과 해제, 세션 메시지 전송 등
  - 두 응용 프로세스 간 세션 설정 : 하나의 응용 프로세스는 동시에 여러 세션 프로세스와 세션 연결 생성 가능
- 동기Synchronization 기능
  - 통신 양단에서 서로 동의하는 논리적인 동기점을 지정하기 위해 사용
  - 오류 복구를 위하여 필수적으로 요구됨
  - 동기점 설정 이전까지는 서로 처리가 완료되었음을 합의했다는 의미
- 대화Dialogue 단계
  - 메시지 전송 과정을 의미
  - 시간 경과에 따라 순차적으로 동기점을 부여해 신뢰성 보장 기능을 단계적으로 구현할 수 있음

02_세션 계층

❖ 토큰

- 두 응용 프로세스의 대화를 관리
- 토큰 보유는 토큰에 부여된 특정한 권리를 배타적으로 소유한다는 의미

▪ 토큰 종류

- 데이터 토큰 : 데이터를 전송할 수 있는 권리 제공
  - 데이터 전송을 위해서는 반드시 데이터 토큰 획득 필요
  - 데이터 토큰을 하나만 사용하면 통신 양단 중 한쪽에서만 데이터 전송 가능(반이중 전송)
- 해제 토큰 : 통신 양단 간의 연결 해제 과정을 제어하기 위해 사용
  - 임의의 사용자가 연결을 해제하려면 해제 토큰 획득 필요
- 동기 토큰 : 세션 연결을 사용하는 과정에서 동기 처리가 필요한 지점에 사용

▪ 토큰과 동기점

- 동기점의 부여 : 큰 파일을 작은 파일로 구분하는 과정
  - 큰 파일을 전송하는 과정의 중간 중간에 동기점을 부여
  - 동기점이 부여되었다는 의미는 해당 내용까지는 전송이 완료되었다는 의미
  - 따라서 전송 과정에 오류가 발생해도 동기점 이후의 과정만 복구하면 됨
- 주동기 토큰Major Token 혹은 Activity Token : 특정 대화 단위를 구분
- 부동기 토큰Minor Token : 대화 단위를 세분화시켜서 구분

02_세션 계층

❖ 동기

- 세션 연결을 사용해 데이터를 주고 받다가 오류가 발생시, 이를 효과적으로 복구할 수 있게 해줌
- 데이터를 전송할 때 특정 지점에서 복구할 수 있도록 양단 간의 합의로 동기점 지정
- 세션 계층의 상위 계층에는 적절한 구간으로 나뉜 지점에 동기점 부여
- 오류가 발생하면 해당 지점으로 돌아가 복구
- 재동기Resynchronization : 동기점을 이용한 일련의 복구 과정

▪ 재동기 기능

- 주동기점
  - 주동기점이 설정된 곳은 완벽히 처리되었다는 의미
  - 주동기점 이전의 복구 과정은 불필요
- 부동기점
  - 복구에 필요한 백업 정보가 상대적으로 적음
  - 오류 복구가 완벽히 이루어지지 않을 수 있음
  - 이전 부동기점에서 복구가 불가능하면 직전 부동기점으로 이동하는 과정을 반복

02_세션 계층

- 동기점의 역할

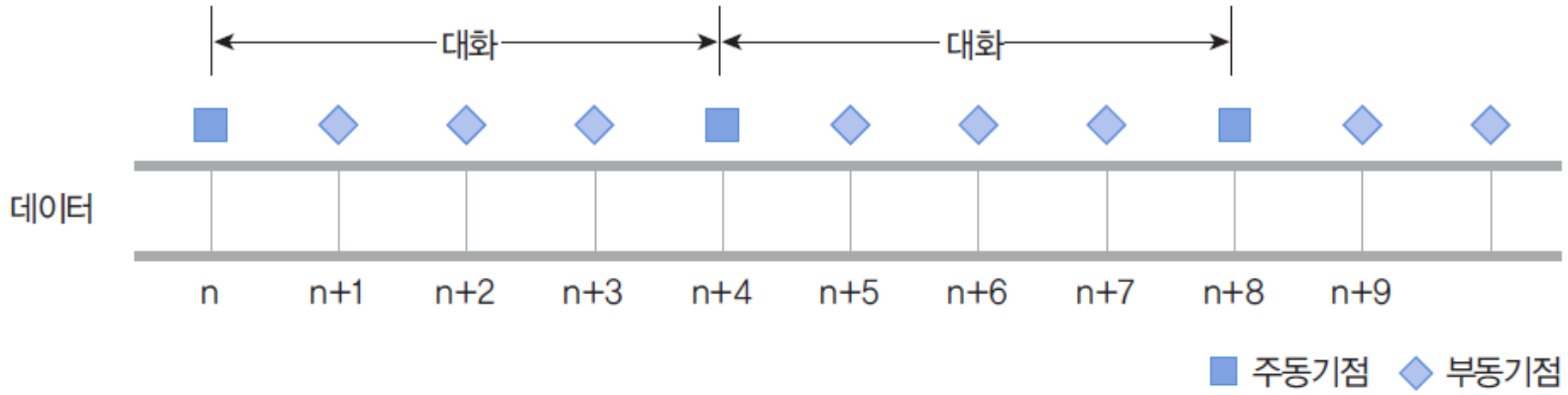


그림 11-2 동기점의 역할

- 액티비티Activity 기능
  - 세션 프로세스 사이에 논리적으로 설정되는 단위, 상호 독립적
    - 예) 파일 여러 개를 전송할 때 각 파일은 하나의 액티비티로 처리되며, 처리 과정에서 주동기점과 부동기점이 부여됨
  - 액티비티 단위의 시작과 끝의 표시는 주동기점의 설정과 동일한 효과를 가짐

02_세션 계층

❖ 세션 연결

- 상위 계층의 응용 프로세스가 다른 응용 프로세스에 세션을 설정하기 위해 사용
- 초기 연결 과정에서 CONNECT 요구가 발생하면 세션 계층은 이를 전송 계층 프리미티브인 CONNECT.request로 변환하여 전송 계층을 통해 상대편 세션 사용자에게 전달

02_세션 계층

❖ 세션 연결

- 다중 세션 연결을 지원하는 서버
  - 하나의 서버 프로세스가 다수의 클라이언트를 동시에 지원
    - 클라이언트 프로세스와 설정된 세션은 논리적으로 연관이 없는 서로 독립적인 연결
  - 서비스 시간이 짧은 경우에 유리
  - 서비스 시간이 길면 클라이언트의 대기 시간이 무한정 증가할 수 있음

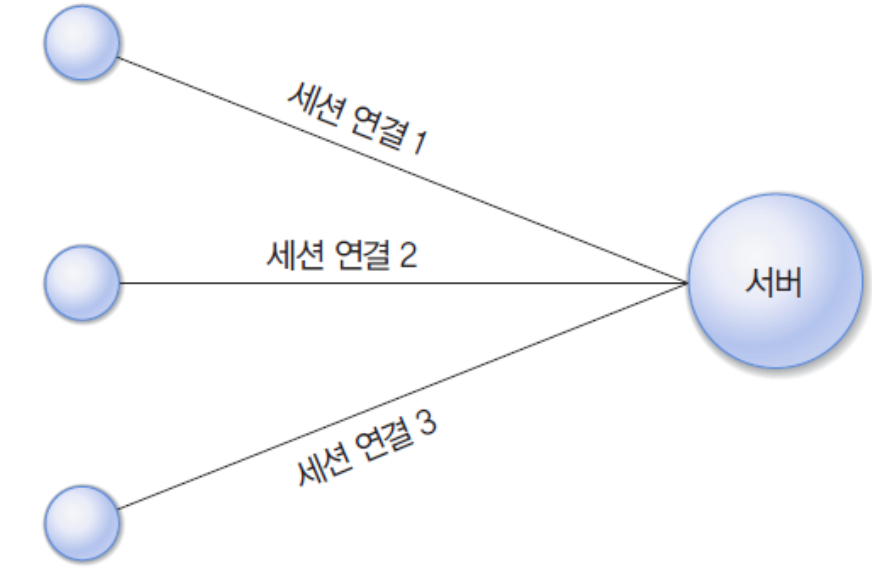


그림 11-3 다중 세션 연결을 지원하는 서버

02_세션 계층

- 단일 세션 연결을 지원하는 서버
  - 하나의 서버 프로세스가 하나의 클라이언트만 지원
  - 대표 서버
    - 클라이언트의 연결 요청을 처리
    - 하위 서버 프로세스를 생성
    - Well-known 포트
  - 단점
    - 개별 요구마다 하위 프로세스를 생성하기 때문에 초기 서비스 환경 구축에 따른 오버헤드가 증가함

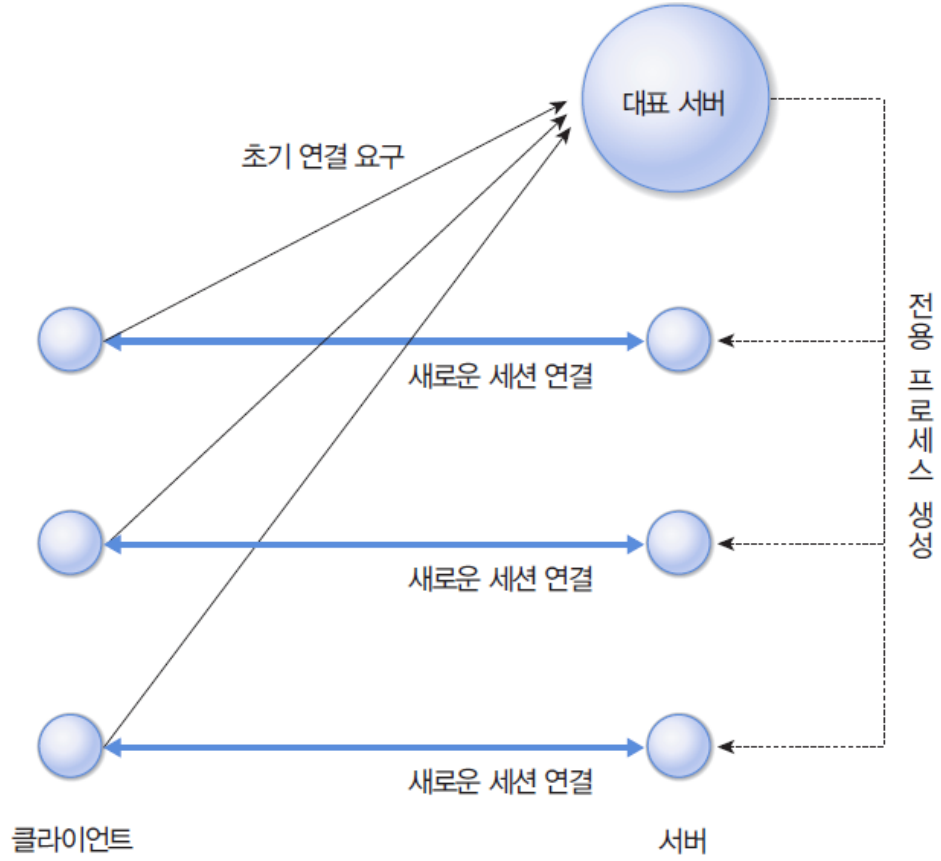


그림 11-4 단일 세션 연결을 지원하는 서버

03_표현 계층

- 응용프로세스 사이에 전송되는 메시지의 표현방법을 다룸
- 표현 계층 프로토콜의 전송 메시지에 표현된 문법Syntax 내용을 통신 양단의 프로세스가 해석하는 기능 제공
- 송신 프로세스가 전달하는 의미Semantic를 수신 프로세스에서 정확히 이해할 수 있게 함

❖ 데이터 표현

- 응용 환경에서 데이터를 표현하는 방법은 컴퓨터마다 다름
  - 예) 문자를 표현하는 방법은 ASCII 코드와 EBCDIC 코드를 사용으로 나뉨
- 서로 다른 코드를 사용하는 컴퓨터끼리 통신하려면 문자 코드 변환 과정 필요

03_표현 계층

- 추상 문법
  - 추상 문법Abstract Syntax : 컴퓨터에서 사용하는 데이터 표현 규칙
  - 전송 문법Transfer Syntax : 네트워크 전체에서 일관성을 지니는 표현 규칙

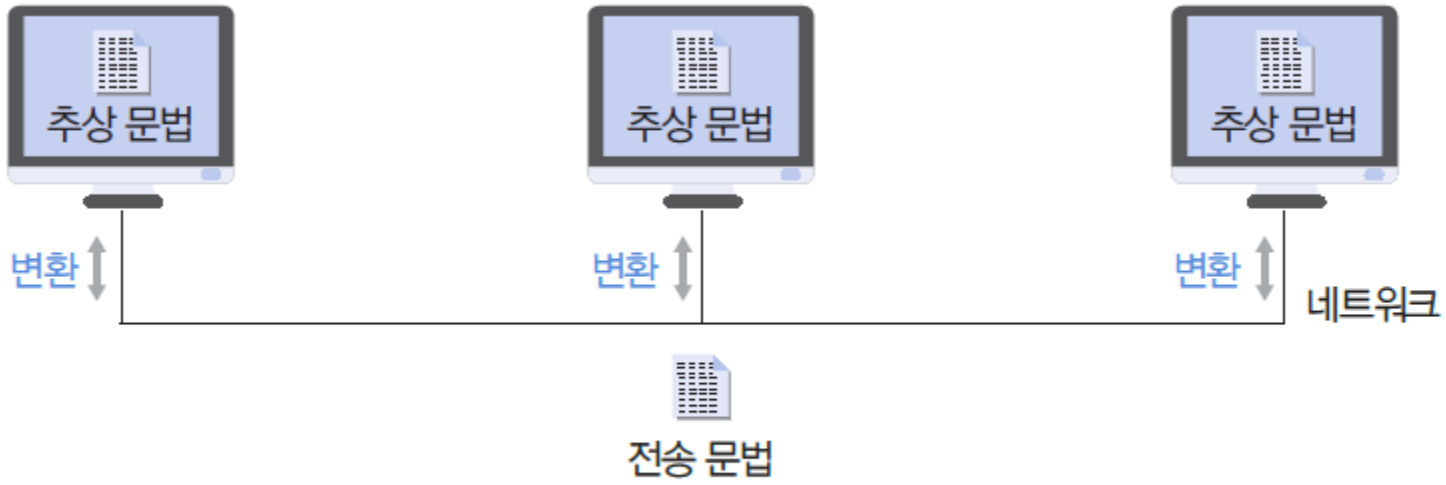


그림 11-5 추상 문법과 전송 문법

03_표현 계층

- ASN.1 Abstract Syntax Notation Number 1
  - 분산 환경에서 표현되는 데이터를 정의하기 위한 일반적인 추상 문법
  - 응용 계층의 문법 규칙뿐만 아니라 PDU Protocol Data Unit의 구조 정의

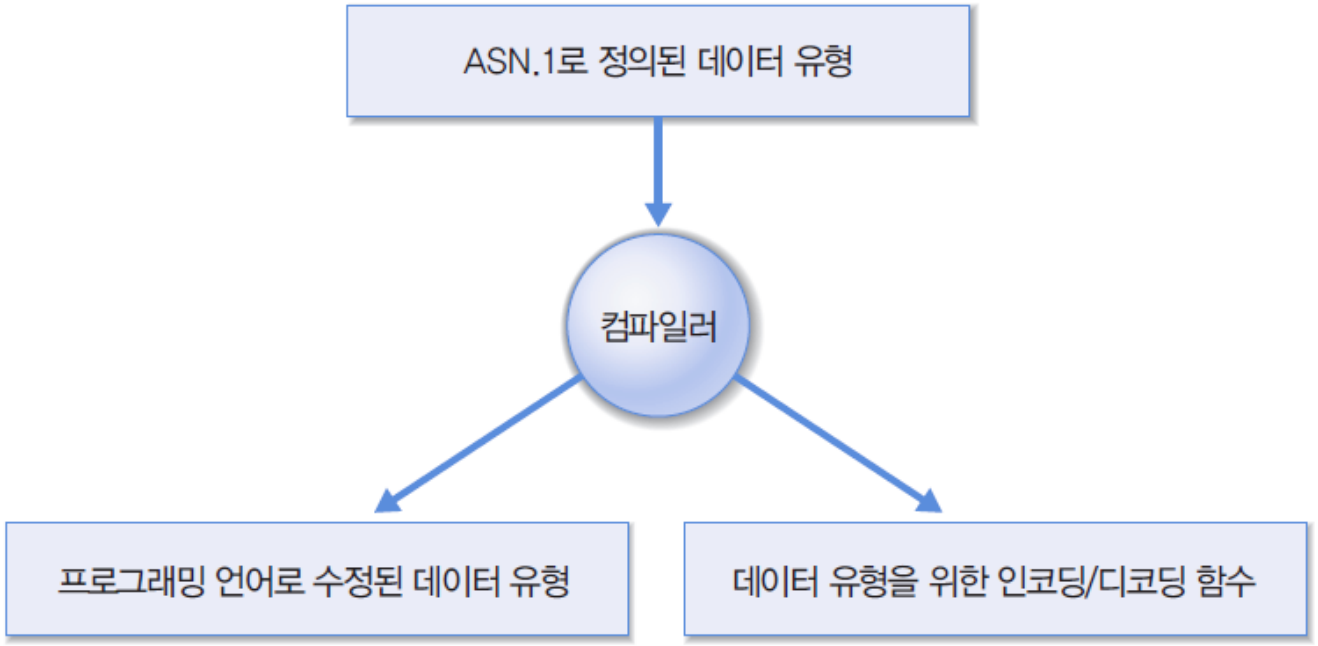


그림 11-6 ASN.1 컴파일러

03_표현 계층

- ASN.1의 기본 목적은 변수 선언과 관련된 데이터 형 정의
- 일반 프로그래밍 언어에서 사용하는 방식과 비슷한 문법 형식 지원
- ASN.1 규약에서 지원하는 클래스 유형

표 11-1 ASN.1 규약에서 지원하는 클래스 유형

유형	설명
UNIVERSAL	일반 데이터형
CONTEXT-SPECIFIC	특정 컨텍스트Context와 관련된 유형
APPLICATION	응용 개체의 공통 유형
PRIVATE	사용자 정의 유형

03_표현 계층

- ASN.1의 UNIVERSAL 클래스
  - 일반 데이터형을 정의
  - 기본형Primitive , 구조형Constructured

표 11-2 UNIVERSAL 클래스의 일반 데이터형 : 기본형

변수형	의미
BOOLEAN	참, 거짓
INTEGER	정수형 숫자
BITSTRING	비트들의 연속
OCTSTRING	옥텟들의 연속
IA5String/GraphString	–
NULL	변수형 미정의
ANY	변수형을 다른 곳에서 정의

  - 기본형을 사용해 유형 정의 예

```
StudentNumber ::= INTEGER
Name = IA5String
MaleFemale ::= BOOLEAN
EntranceRight ::= BITSRING {accept(0), reject(1)}
PersonalInformation ::= OCTETSTRING
```

03_표현 계층

- UNIVERSAL 클래스의 일반 데이터형 : 구조형

변수형	의미
SEQUENCE	순서대로 나열한 임의 유형의 집합(고정 크기)
SEQUENCE OF	순서대로 나열한 동일 유형의 집합(고정/가변 크기)
SET	순서 없이 나열한 임의 유형의 집합(고정 크기)
SETOF	순서 없이 나열한 동일 유형의 집합(고정/가변 크기)
CHOICE	미리 정의된 유형 집합에서 선택한 순서가 없는 유형 집합(고정 크기)

- 구조형을 사용한 예

```
Student Record ::= SEQUENCE {
        StudentNumber INTEGER,
        Name IA5String,
        MaleFemale BOOLEAN
}
```

03_표현 계층

- ASN.1의 태그
  - 태깅Tagging : 구조형에 선언된 변수를 개별적으로 사용
  - 태그 선언
    - CONTEXT-SPECIFIC : 태그의 범위가 현재 구조형에 한정되어 적용됨
    - APPLICATION : 태그의 범위가 전체 응용 컨텍스트에 적용됨
    - PRIVATE : 태그의 범위가 해당 사용자에게 적용됨
  - 태그 사용 예

```
Student Record ::= SEQUCNE {
        StudentNumber [APPLICATION 1] INTEGER,
        Name [1] IA5String
        MaleFemale [2] BOOLEAN
}
```

- 데이터 압축과 보안
  - 압축 : 전송 데이터의 양을 줄이는 목적으로 사용
  - 암호화 : 전송 데이터의 내용을 해석하지 못하도록 하기 위하여 사용

03_표현 계층

❖ 데이터 압축

- 대용량 데이터는 압축하여 크기를 줄인 후 전송하는 것이 속도에서 유리함
- 데이터의 특성에 맞는 알고리즘을 사용하는 것이 중요함

▪ 연속 문자 압축의 예

- 알고리즘

```
struct send_data {      /* 그림 크기를 18×10으로 가정, 크기는 3바이트 */
        char pattern;   /* 원래 데이터에 포함된 임의의 패턴 */
        short count;    /* pattern에 보관된 데이터의 반복 횟수 */
}
```

- 데이터 압축 예
  - T라는 문자를 모자이크 형태로 형상화한 정지 영상 데이터
  - 원본 데이터 가로 18바이트, 세로 10바이트 크기 : 18×10=180바이트
  - 압축 데이터 크기는 3×19=57바이트, 원본 데이터보다 180-57=123바이트 줄임
  - 압축 해제 : 18×10의 밑그림 위에 pattern의 내용을 count만큼씩 표시함

03_표현 계층

❖ 데이터 압축

- 대용량 데이터는 압축하여 크기를 줄인 후 전송하는 것이 속도에서 유리함
  - 인터넷 환경의 대용량 멀티미디어 데이터, 대용량 실시간 데이터 등
- 압축 정도는 데이터 패턴에 영향 받음 : 중복 데이터가 많으면 높은 압축률
- 데이터의 특성에 맞는 알고리즘을 사용하는 것이 중요함

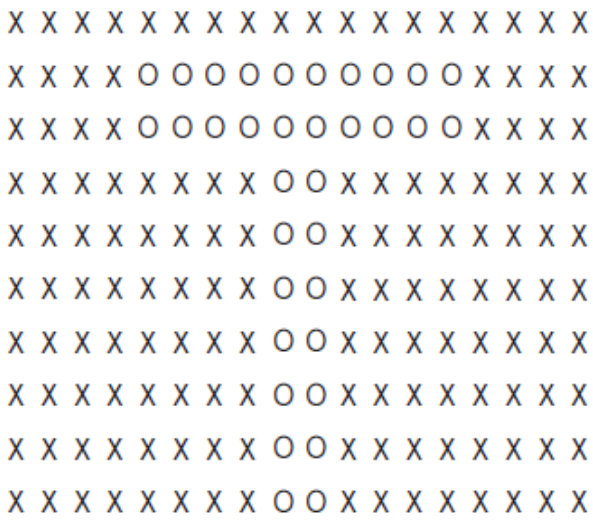
▪ 연속 문자 압축의 예

- 알고리즘

```
struct send_data {      /* 그림 크기를 18×10으로 가정, 크기는 3바이트 */
        char pattern;   /* 원래 데이터에 포함된 임의의 패턴 */
        short count;    /* pattern에 보관된 데이터의 반복 횟수 */
}
```

03_표현 계층

- 데이터 압축 예
  - T라는 문자를 모자이크 형태로 형상화한 정지 영상 데이터
  - 원본 데이터 가로 18바이트, 세로 10바이트 크기 : 18×10=180바이트
  - 압축 데이터 크기는 3×19=57바이트, 원본 데이터보다 180-57=123바이트 줄임
  - 압축 해제 : 18×10의 밑그림 위에 pattern의 내용을 count 만큼씩 표시함



(a) 원본 데이터

pattern	count
X	22
O	10
X	8
O	10
X	12
O	2
X	16
O	2
X	16
O	2
X	16
O	2
X	16
O	2
X	16
O	2
X	8

(b) 압축 데이터

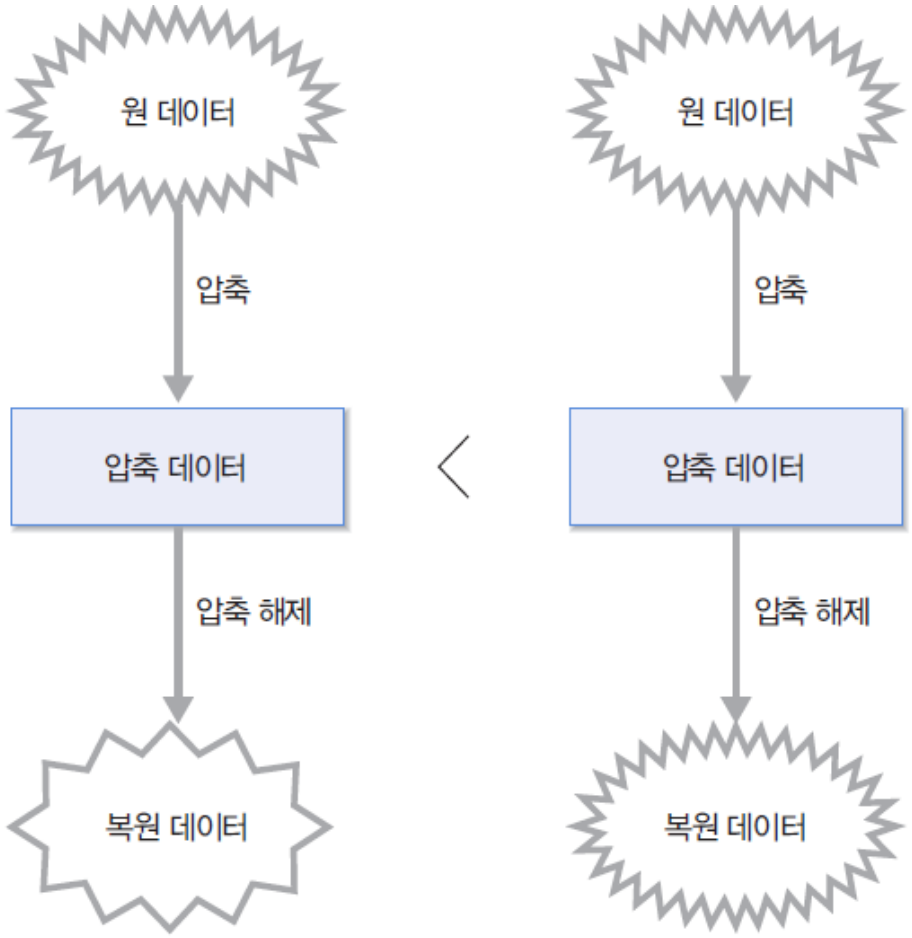
그림 11-7 데이터 압축 예

03_표현 계층

- 손실·비손실 데이터 압축
  - 비손실 압축
    - 압축 과정에서 원래 데이터의 내용을 분실하지 않음
    - 압축 해제 과정을 통해 원래의 데이터를 100% 복원
  - 손실 압축
    - 압축 과정에서 원래 데이터의 내용을 부분적으로 분실
    - 압축 해제 과정을 통해 원래의 데이터를 100% 복원할 수 없음
    - 압축 효율을 높이기 위하여 사용

03_표현 계층

- 손실·비손실 압축



(a) 손실 압축 (b) 비손실 압축

그림 11-8 **손실 · 비손실 압축**

04_응용 계층

- 하부 계층을 이용해 사용자에게 편리한 응용 환경 제공
  - 하부 계층들의 기본 역할은 신뢰성 있는 데이터 전송 보장

❖ 클라이언트-서버 모델

- 서버가 먼저 통신 대기 상태, 비대칭 구조는 클라이언트와 서버의 연동을 단순화 시키는 장점이 있음

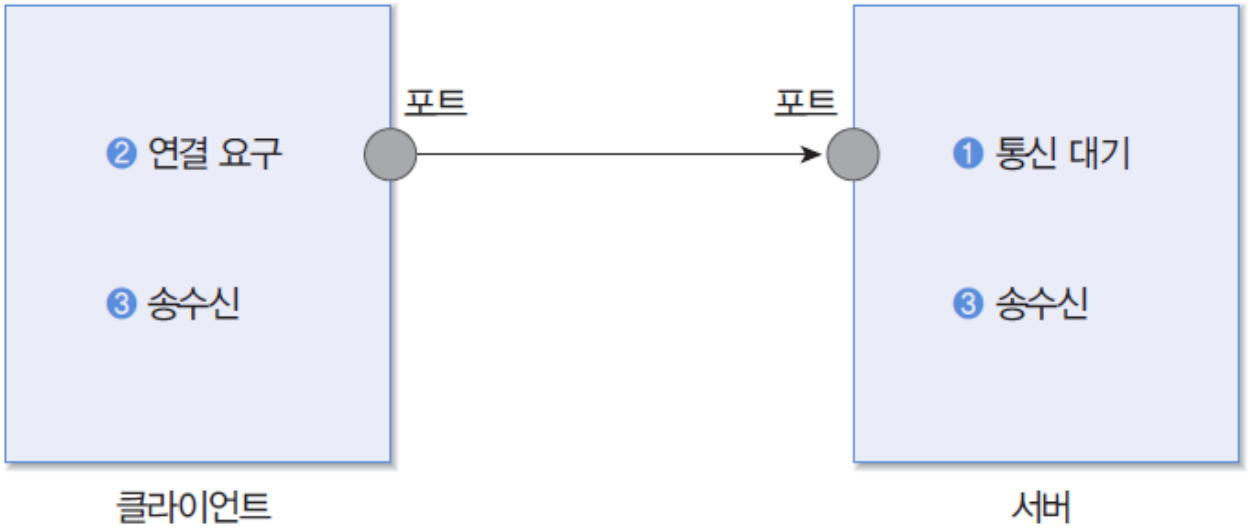


그림 11-9 클라이언트와 서버의 연결

04_응용 계층

- 연결형·비연결형 서비스
  - UDP : 속도는 빠르지만, 신뢰성에 문제가 있음(데이터 분실, 비순서 도착)
  - TCP : 신뢰성이 높지만, 상대적으로 속도가 느림
- 상태 정보
  - 상태 : 특정 상황에 대하여 통신 양단이 바라보는 관점
  - 오류 발생시 서로 합의할 수 있는 상태로 되돌아가는 과정이 복구 과정임
  - 비상태 서비스
    - 상태가 없으므로 복구 과정이 간단함
    - 파일 공유 서비스는 대표적인 비상태 서비스의 예
- 동시성 제어
  - 동시성
    - 임의의 여러 동작이 외형상 동시에 진행되는 것처럼 보이는 현상
    - 여러 동작의 선후 진행 속도가 실행 결과에 영향을 미치지 않음
    - 예 : 단일 CPU가 장착된 컴퓨터에서 여러 프로세스가 실행되는 경우
  - 동시성 제어
    - 독립적으로 실행되는 프로세스의 실행 순서가 결과에 영향을 주지 않음

Thank You