



1

## 구조적 분석기법 개요

# 01 구조적 분석기법 개요

## 1 1970년대 탄생한 분석기법

 사용자의 필수적 요구사항을 적절한 기능단위로 분할하고 개발흐름에 따라 소프트웨어를 모델링하는 분석기법

-  대표적 구조적 분석기법
- 기능 중심의 자료흐름도(DFD)
  - 자료사전(DD)
  - 소단위명세서(Mini Spec)

# 01 구조적 분석기법 개요

## 2 구조적 분석기법의 원칙



### 추상화의 원칙

- 🔍 특정 대상에 대한 실체를 분리하기 위하여  
‘어떻게’가 아닌 ‘무엇’으로 표현하는 간소한 방법
- 🔍 사소한 것에 제약을 받지 않고 문제를  
해결할 수 있게 함

# 01 구조적 분석기법 개요

## 2 구조적 분석기법의 원칙


### 정형화의 원칙

-  소프트웨어의 제어와 산출물의 품질관리를 위한 기초가 됨
-  형식이 생각과 명령을 자동화시킬 수 있는 근거를 제공

# 01 구조적 분석기법 개요

## 2 구조적 분석기법의 원칙

### 분할정복

-  복잡하고 큰 시스템을 좀 더 작고 독립적인 서브시스템으로 나누고, 작게 분할된 시스템들을 쉽게 개발 및 해결하고자 하는 개념

# 01 구조적 분석기법 개요

## 2 구조적 분석기법의 원칙

### 계층적 구조

- 🔍 분할정복을 위해 나누어진 모듈들을 상호 연관 관계 및 구조에 대한 이해도 향상에 도움이 될 수 있도록 함
- 🔍 의사소통과 제어 문제를 다룰 수 있음

# 01 구조적 분석기법 개요

## 2 구조적 분석기법의 원칙

### 계층적 구조

- 🔍 분할정복을 위해 나누어진 모듈들을 상호 연관 관계 및 구조에 대한 이해도 향상에 도움이 될 수 있도록 함
- 🔍 의사소통과 제어 문제를 다룰 수 있음

# 01 구조적 분석기법 개요

## 3 모듈화

### 다익스트라

- 🔍 구조적 프로그래밍(Structured Programming)
- 🔍 GOTO문이 없는 프로그램의 개발 주장
- 🔍 프로그램의 복잡도를 낮추기 위하여 모듈 단위로 프로그램을 분할



# 01 구조적 분석기법 개요

## 3 모듈화

### 콘스탄틴

- 🔍 모듈러 프로그래밍(Modular Programming)
  - 프로그램의 독립성을 강조
  - 기능과 서브루틴으로 모듈화
- 🔍 DFD의 발전에 공헌
- 🔍 소프트웨어의 인간적인 측면을 강조

# 01 구조적 분석기법 개요

## 4 DFD의 발전

### 요든

- 🔍 요든 구조적 방법 (YSM : Yourdon Structured Method)
  - : 기능구조를 중요시
    - Feasibility study
    - Essential modeling
    - Implementation modeling
- 🔍 DFD의 발전에 공헌

# 01 구조적 분석기법 개요

## 4 DFD의 발전

### 디마르코

- 🔍 구조적 분석 및 명세화(Structured Analysis & System Specification)방법
- 🔍 DFD(Data Flow Diagram)의 발전에 공헌

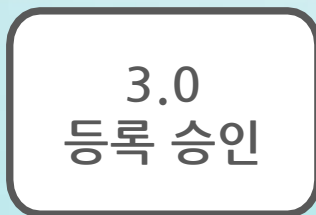
## 2 DFD 작성

## 02 DFD 작성

### 1 DFD 작성을 위한 기호

#### ① 프로세스

- 🔍 전환되거나 저장되거나 분배되기 위한 데이터에 수행되어 지는 작업 또는 행동을 표현함
- 🔍 모서리가 둥근 사각형으로 표시
- 🔍 프로세스의 이름과 번호가 기록됨



[프로세스 예]

## 02 DFD 작성

### 1 DFD 작성을 위한 기호

#### ② 데이터 저장소

- 데이터를 표현
  - 파일 폴더(File Folder)
  - 컴퓨터 파일
  - 노트북
- 오른쪽 수직선이 없는 사각형으로 작성
- 숫자와 저장소의 이름을  
표찰(Label)로 작성

D1: 과목 파일

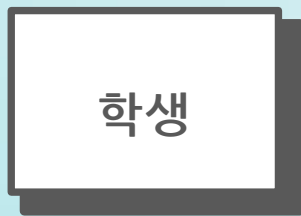
[데이터저장소 예]

## 02 DFD 작성

### 1 DFD 작성을 위한 기호

#### ③ 외부 엔터티

- 🔍 데이터의 원천과 목적지를 기술
- 🔍 때때로 외부 개체(External Entity)로 불려짐
- 🔍 사각형 기호로 표시
- 🔍 이름은 외부 에이전트가 무엇인지를 설명
- 🔍 분석 대상의 외부에 존재하기 때문에, 구체적인 특징은 분석에 무관함



[외부엔터티 예]

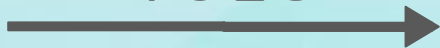
## 02 DFD 작성

### 1 DFD 작성을 위한 기호

#### ④ 데이터 흐름

- 이동하고 움직이는 데이터의 흐름을 표현
- 화살표로 표시
- 데이터를 표현하기 위해서 의미 있는 이름을 선택하여 화살표 주변에 기술

수강신청



[데이터 흐름 예]



## 02 DFD 작성

### 2 DFD 작성방법

#### ① 배경도

- 🔍 개발하게 될 시스템의 명칭을 부여  
⇒ 수강신청시스템
- 🔍 시스템과 관련된 인터페이스를 정의



※ 출처 : 실무에 바로 활용하는 소프트웨어공학, 김희영저, 21세기사, 2018

## 02 DFD 작성

### 2 DFD 작성방법

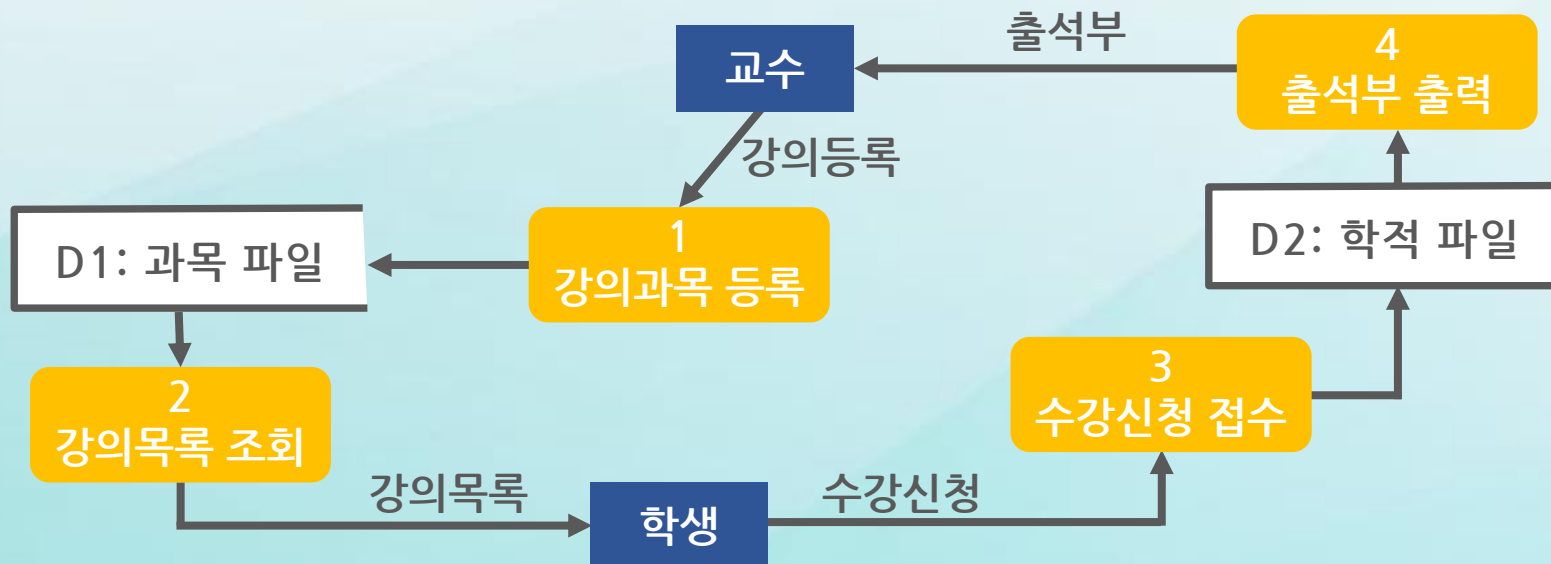
#### ② 0 수준 DFD를 위한 인터뷰

- 🔍 구체적인 수강신청 프로세스를 DFD로 표현하기 위해 시스템분석가는 학생과 교수에게 인터뷰를 진행함
- 🔍 인터뷰 진행결과 구체적인 DFD를 작성할 수 있게 되었으며 학생과 교수를 통해 확인 및 검증이 가능함

## 02 DFD 작성

### 2 DFD 작성방법

#### ③ 0 수준 DFD를 위한 인터뷰 결과



※ 출처 : 실무에 바로 활용하는 소프트웨어공학, 김희영저, 21세기사, 2018

## 02 DFD 작성

### 2 DFD 작성방법

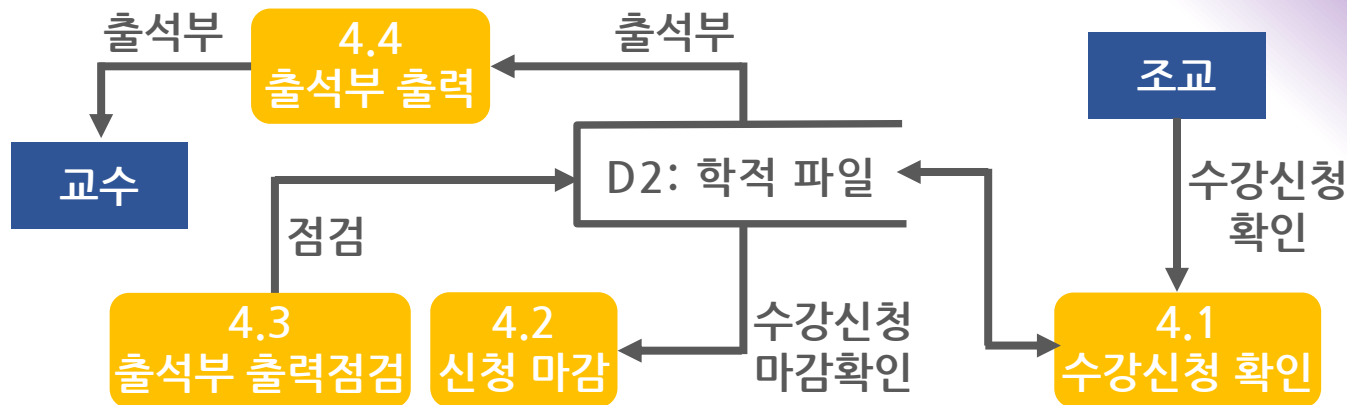
#### ④ 1 수준 DFD를 위한 상세화 작업

- 🔍 최종적으로 프로그램 구현이 가능한 수준까지 DFD는 상세화 될 수 있음
- 🔍 더 이상 구체화되기 어려운 수준으로 정의된 이후 명세서를 작성하여 내용을 서술함

## 02 DFD 작성

### 2 DFD 작성방법

#### ⑤ 1 수준 DFD를 위한 상세화 작업 결과 예



※ 출처 : 실무에 바로 활용하는 소프트웨어공학, 김희영저, 21세기사, 2018

## 02 DFD 작성

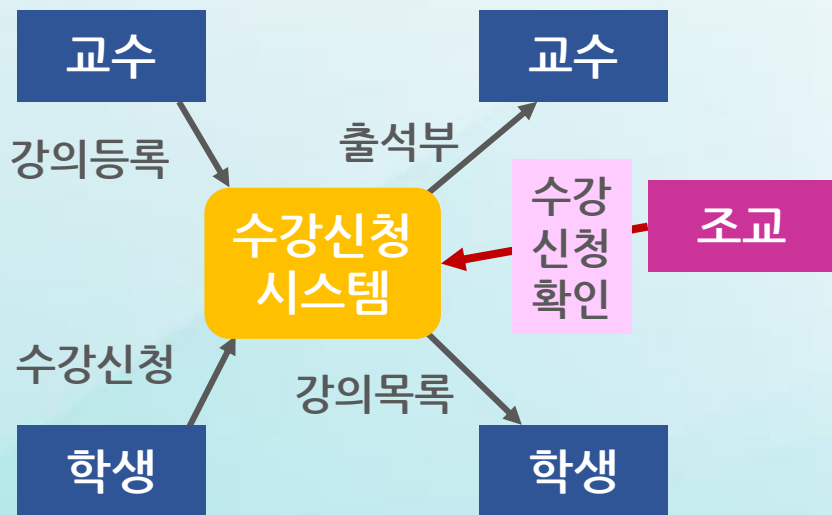
### 3 DFD 작성시 주의사항

#### ① 상위수준에서 하위수준으로 일관성 유지

- 🔍 새로운 외부 엔터티 추가 시  
⇒ 배경도 수정
- 🔍 더 이상 구체화할 수 없는 수준의 DFD  
⇒ 소단위 명세서

#### ② 자료저장소의 구체화

- 🔍 자료저장소는 자료사전으로 표현되어야 함
- 🔍 관계형 데이터베이스를 고려하여 구체화 함



※ 출처 : 실무에 바로 활용하는 소프트웨어공학,  
김희영저, 21세기사, 2018

## 02 DFD 작성

### 4 DFD 작성지침

#### ① 모든 DFD에 적용되는 규칙

- 프로세스로 들어가는 입력 데이터는  
그 프로세스로부터 나오는 출력 데이터와 달라야 함
- DFD의 각 기호는 유일한 명칭을 가져야 함

## 02 DFD 작성

### 4 DFD 작성지침

#### ① 모든 DFD에 적용되는 규칙



※ 출처 : 실무에 바로 활용하는 소프트웨어공학, 김희영저, 21세기사, 2018



## 02 DFD 작성

### 4 DFD 작성지침

#### ② 기호별 작성규칙



##### 프로세스 규칙

- 출력만 있는 프로세스는 있을 수 없음
- 입력만 있는 프로세스는 있을 수 없음
- 프로세스의 이름은 동사형의 작명규칙을 가짐

## 02 DFD 작성

### 4 DFD 작성지침

#### ② 기호별 작성규칙



##### 데이터저장소 규칙

- 한 데이터 저장소로부터 다른 데이터 저장소로 직접 이동할 수 없음
- 외부 엔터티로부터 데이터 저장소로 직접 이동할 수는 없음
- 데이터 저장소에서 외부엔터티로 직접 이동할 수도 없음
- 데이터 저장소의 명칭은 명사형으로 작성되어야 함

## 02 DFD 작성

### 4 DFD 작성지침

#### ② 기호별 작성규칙



##### 외부 엔터티 규칙

- 데이터는 외부 엔터티에서 외부 엔터티로 직접 이동할 수 없음
- 외부 엔터티의 이름은 명사형으로 작성되어야 함

## 02 DFD 작성

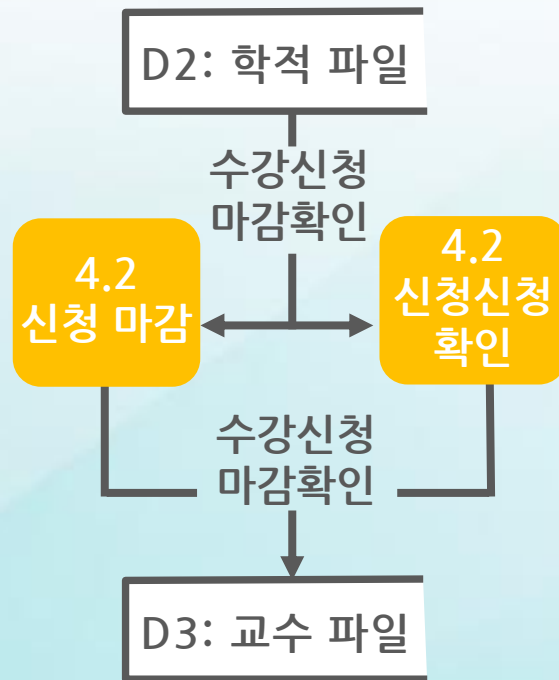
### 4 DFD 작성지침

#### ② 기호별 작성규칙



#### 데이터 흐름 규칙

- 데이터 흐름은 기호 사이에 오직 한 방향의 흐름을 가짐
- 한번 지나간 동일한 프로세스로 직접 역행이 불가능
- 데이터 저장소로의 데이터 흐름은 갱신을 의미
- 데이터 저장소로부터 데이터 흐름은 추출이나 조회를 의미
- 데이터 흐름의 이름은 명사형으로 작성되어야 함
- 데이터 흐름은 분기 혹은 결합이 가능



※ 출처 : 실무에 바로 활용하는 소프트웨어공학,  
김희영저, 21세기사, 2018

## 02 DFD 작성

### 4 DFD 작성지침

#### ③ DFD의 기능적 분해

- 🔍 완전성
  - 모든 구성요소를 포함하여야 함
  - 데이터는 자료저장소에 완전히 기술되어야 함
- 🔍 균형성
  - 상하위 수준간에 입력과 출력은 일관성을 가져야 함
- 🔍 반복적 개선
  - 시스템 분석가는 여러 번 반복해서 DFD를 수정 보완하면서 개선해야 함

## 02 DFD 작성

### 4 DFD 작성지침

#### ③ DFD의 기능적 분해

- 🔍 타이밍
  - 시스템이 가동되지도 않고, 멈추고 있지도 않은 것처럼 작성
- 🔍 분해의 중단
  - 프로세스를 더 이상 지속적으로 분해할 수 없는 시점이 오게 됨

## 02 DFD 작성

### 5 DFD 작성순서

1	배경도 작성	전체 시스템과 관련된 외부 엔터티를 정의
2	DFD 1차분할	기능구성의 큰 단위로 프로세스를 정의
3	DFD 2차분할	데이터 저장소와 관련 프로세스를 구체화
4	DFD 검토	프로세스에 대한 상세한 기능 정의 및 데이터저장소와 연계
5	요구사항 명세서 첨부	요구사항 명세서에 DFD를 추가

### 3 자료사전 작성



## 03 자료사전 작성

### 1 자료사전 작성



자료사전은 DFD의 자료저장소에 대한 구체적인 내용을 설명하기 위해 작성함

- ① 자료흐름을 구성하는 자료항목 정의
- ② 자료저장소를 구성하는 자료항목 정의
- ③ 자료에 대한 의미 정의
- ④ 자료원소의 단위 및 값 정의

## 03 자료사전 작성

### 2 자료사전의 사용기호

기호	의미
=	정의 (Is Composed Of)
+	구성 (And, Along With)
{ }	반복 (Iteration Of)
[ ]	택일 (Choose Only One Of)
( )	생략가능 (Optional)
**	주석 (Comment)

## 03 자료사전 작성

### 3 자료사전의 예시

- 🔍 졸업예정자정보=소속+학번+지도교수+입학년도
- 🔍 소속 = [“주”|“야”] + (단과대)+학과 + 학년
- 🔍 학번=입학년도+단과대정보두자리+일련번호
- 🔍 학과=\*전체 5자리 이하로 표현\*
- 🔍 지도교수=이름+전화번호
- 🔍 입학년도=\*숫자 4자리”

4

## 소단위 명세서 작성

## 04 소단위 명세서 작성

### 1 소단위 명세서 개요

- 🔍 소단위 명세서는 분석결과를 바로 구현하기에 부족하기 때문에 작성
- 🔍 소단위 명세서는 분석과 구현의 중간과정으로 활용하기 위해 작성함
- 🔍 더 이상 구체화되지 않는 프로세스 최소단위에서 프로세스에 대한 구체적인 설명을 글로 서술함
- 🔍 화면 혹은 보고서 출력에 대한 처리과정과 처리조건을 글로서 표현
- 🔍 처리로직으로 표현하기도 함

## 04 소단위 명세서 작성

### 2 소단위 명세서 기술방법

- 오직 최하단위 단계의 처리만 기술함
- 항상 간결하고, 명료하며, 완전하고, 의미에 애매모호함이 없도록 기술해야 함
- 배경도와 중간단계의 처리는 다시 자료 흐름도로 정의 되므로 소단위 명세서에 기술하지 않음
- 최하위 단계 외 처리에 대해서도 소단위 명세서를 작성하는 것은 시스템을 중복 명세하게 되어 간단 명료하지 않을 뿐더러 문서의 유지에도 더욱 어렵게 됨

## 04 소단위 명세서 작성

### 3 소단위 명세서 작성 예

```
과목목록=FIND_DB(과목목록, 연도, 학기)
DO WHILE 과목목록의 과목명 중 1개
출석부=FIND_DB(출석부, 연도, 학기, 과목명)
  DO WHILE 출석부의 학생 중 1명
    결석수 = 출석부의 학생 결석수를 더한 수
    성적부의 성적 = 성적부 성적  결석수*2
  END DO
END DO
```