2주차 3차시 설계도구

【학습목표】

1. 순서도 및 자료흐름도의 개념 및 작성시 유의사항을 설명할 수 있다.
2. 구조적 설계도구의 종류 및 각각의 개념과 특징을 구분할 수 있다.

학습내용1 : 순서도

*** 순서도 (Flowchart)**

① 시스템 순서도 (System Flowchart)

② 프로그램 순서도 (Program Flowchart)
- **개략 순서도 (General Flowchart)**
- **상세 순서도 (Detail Flowchart)**

학습내용2 : 자료흐름도

1. 자료흐름도(DFD : Data Flow Diagram) 구성요소

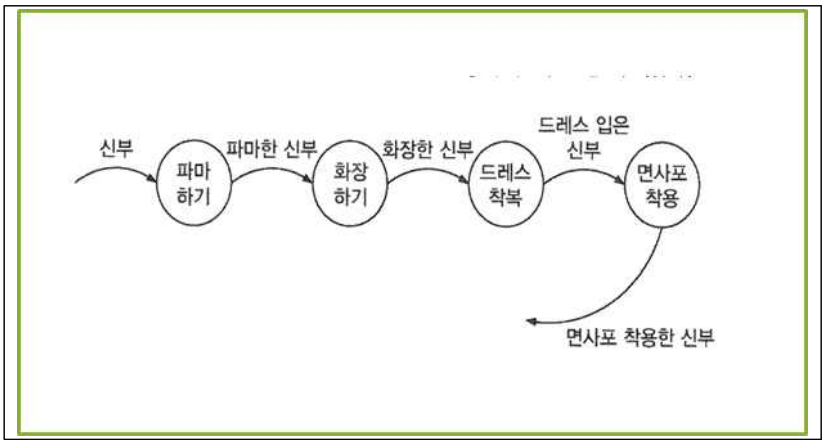
- 교재 「P. 72」의「표 4-4」 참조

구성요소 명칭	표현기호 (Yourdon)	표 현 기 호 (Gane과 Sarson)	의 미
자료 흐름 (data flow)	→	→	단말, 처리, 자료저장소를 연결하여 흐름을 나타낸다.
처 리 (process)	○	□	입력자료의 흐름을 출력자료 흐름으로 변환한다.
자료저장소 (data store)	=	⊏	한 처리에서 다음 처리로 직접 전달되지 않는 자료의 일시적인 저장장소를 의미한다.
단 말 (terminater/ external entity)	□	□	시스템의 외부에 존재하는 사람이나 조직체를 의미한다.

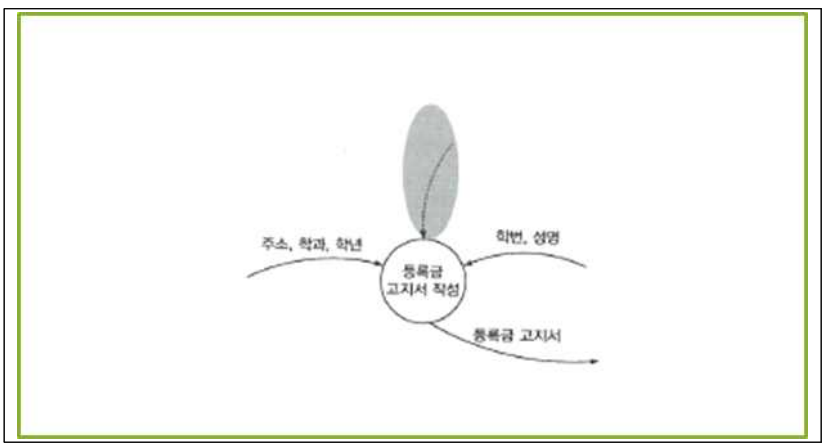
- 요든(Yourdon)의 기호 : 우리나라에서 많이 사용함
- 가네(Gane)와 살손(Salson)의 기호
- 이들 구성요소의 사용상 유의사항에 대해서는 교재「PP. 73~78」참조

1) 자료흐름도 작성 지침
- 하나의 자료흐름도에 항목(Item)의 수준(Level)이 혼합되지 않아야 함
- 「Yourdon의 기호」 또는「Gane와 Salson의 기호」가 혼합해서 사용하면 않됨
- 표준화·문서화를 위해서 자료흐름도는 템플릿(template)으로 작성
- 자료흐름(data flow)으로 모든 기호는 연결되어야 함
- 모든 기호 · 연결자에는 이름을 붙여야 함

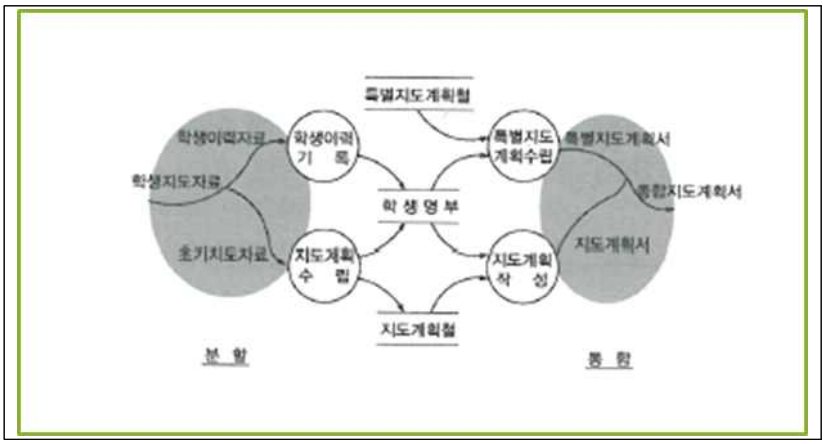
2) 자료흐름도 작성 시 유의사항
- 자료흐름도에서 표현하는 모든 이름은 정확하게 표현해야 함
- 포괄적 이름 사용은 피해야 함
- 이름은 하나의 명령형 동사와 특정 목적어로 명명해야 함
- 입력된 자료흐름이 처리 후에 출력된 자료흐름으로 변환된 경우 ⇨새로운 이름을 부여해야 함



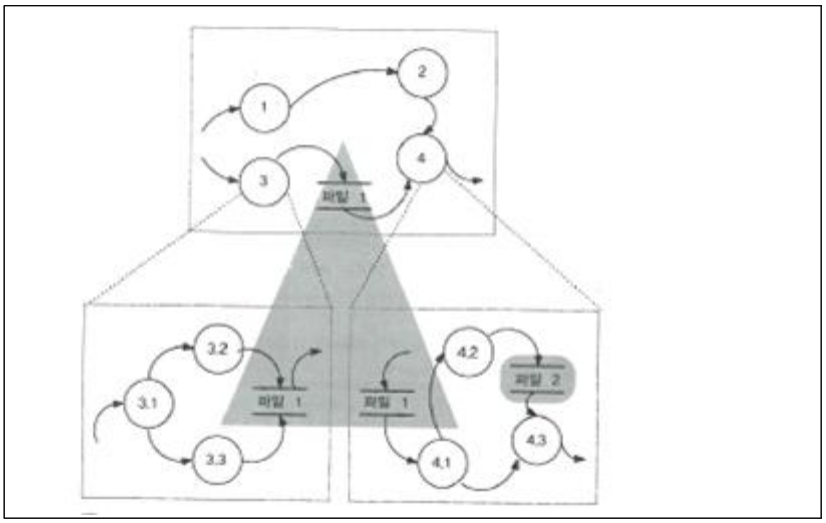
- 자료흐름도에는 자료보존법칙(Data Conservation Rule)이 준수되어야 함

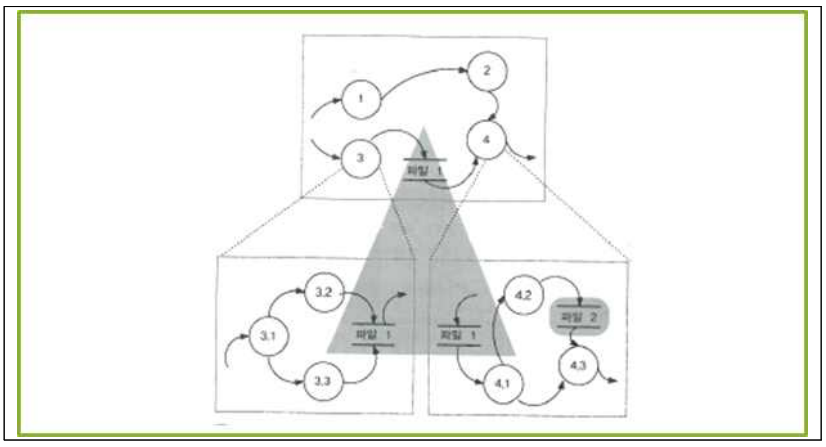


- 자료흐름(Data Flow)은 논리적 모순이 없다면 분할 및 통합되어도 무방함



- 자료흐름도는 자료저장소 균형의 법칙(File Balancing Rule)을 준수해야 함





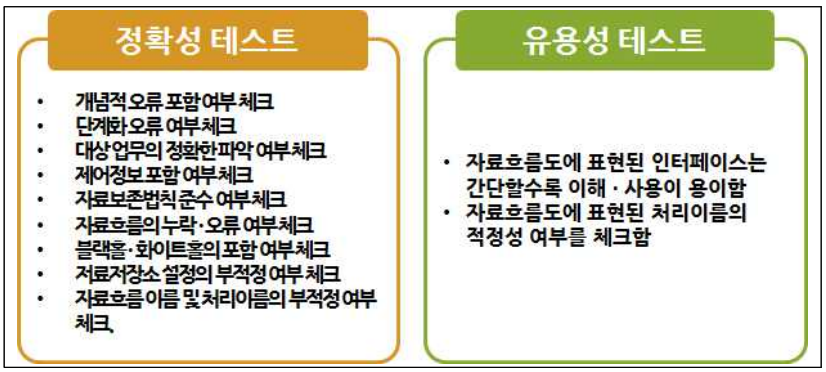
3) 자료흐름도의 단계화
- 작성자의 편의성 제고, 사용자의 이해의 용이성 제고 등을 위하여 단계화 함
- 자료흐름도를 단계화 시키면 최상위도 (Context Diagram), 중위도(Middle Diagram), 최하위도 (배경도 : Functional Primitives)

① 최상위도
- 하나의 시스템에 한 개만 작성
- 한 개의 프로세스 (Process), 1~n개의 단말(Terminator)로 구성

② 중위도
- 시스템 구성의 전체적인 상태를 개략적으로 파악하기 위해 작성하며, 업무의 복잡도에 따라서 1~n개 단계의 중위도가 작성됨
- 시스템의 중요부분을 파악토록 전체적 기능을 개략적으로 구분해서 작성함
- 시스템의 개괄적 이해에 필요함
    - 구체적이고 상세한 사항은 최하위도에서 파악함

③ 최하위도
- 일반적으로 기능 단위로 하나의 최하위도가 작성됨
- 더 이상 하위단계로 분할 불가능한 처리(Process)로 구성된 자료흐름도 임
- 최하위도 분할 원칙
    - A4용지 1매에 미니스팩(mini spec)으로 표현할 분량 정도
    - 하나의 처리(process)에 입출력 흐름이 한 개 정도인 수준
    - 하나의 처리에 「입력 : 출력」의 비율이 「1 : n」정도
    - 자료흐름도에 처리의 개수는 「7±2」정도가 적합함

〈자료흐름도의 평가는 다음과 같은 원칙을 준수해야 함〉
- 정확성 · 유용성을 테스트함
- 평가결과 문제가 제기되어 개선이 필요한 경우 재분할하여 수정 · 보완 · 재작성 해야 함



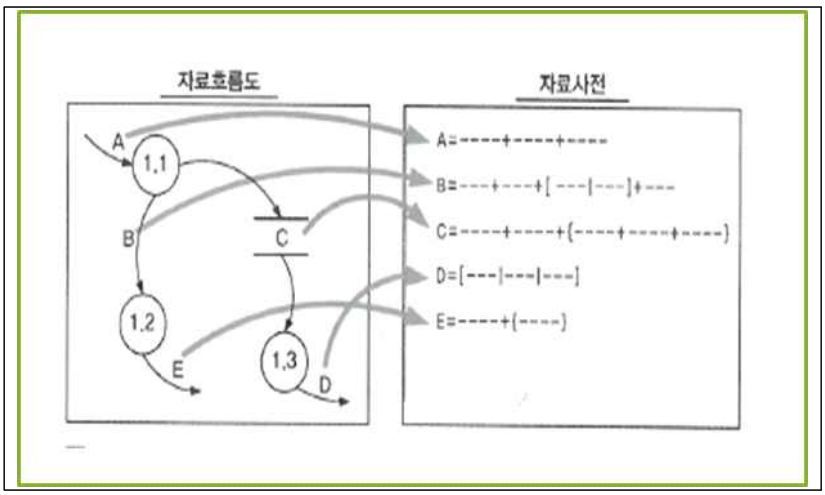
학습내용3 : 구조적 설계 도구

*** 구조적 설계 도구**
- **자료흐름도**
- **자료 사전**
- **미니스팩**
- HIPO Chart
- N-S Chart
- **워니어-오어 차트(Warnier-Orr Chart)**
- **프로그램 기술 언어**

1. 자료사전

1) 자료사전(Data Dictionary : DD)
- 자료흐름도에 표기된 모든 ⓐ 자료흐름, ⓑ 자료저 장소에 대해서 이들을 구성하는 자료항목(Data Item)
- 자료의 의미
- 자료원소의 단위 및 값을 ⇦정의하는 도구(Tool)임

2) 자료흐름도(DFD)와 자료사전(DD)은 상호 보완적인 관계임



3) 자료사전의 정의 내용은 다음과 같음

* 정의 대상
- 자료흐름
- 자료요소
- 자료저장소
- 정의어와 주석

〈구체적인 정의 예 : 교재「PP. 89~91 상단」까지 참조〉

4) 자료사전의 표현방법과 원칙
- 사용 기호(symbol) 교재「P. 85~86」의「표 4-5, 6」참조
- 기호의 사용법과 의미 : 교재「PP. 85~88의 중간」까지 참조

* 자료사전의 기본적 구조
- 순차구조
- 반복구조
- 선택구조 등이 복합구조를 이룸
  - 교재「PP. 88~89」 참조

* 자료사전 정의 시 유의사항
- 자료사전 이름이 중복되면 안됨
- 자료사전 이름으로 쉽게 정의를 찾을 수 있어야 함
- 갱신의 용이성을 충분히 배려해야 함
- 자료 정의 시에 모호성 배제를 위하여 세부 구성 내역을 명세 하는 원칙을 수립해야 함

2. 미니스팩

1) 미니스팩 (Mini Spec : 미니 명세서 : 소단위 명세서 등으로 호칭됨)

2) 미니스팩의 기능
- 자료흐름도 최하위도(Functional Primitives)의 최하위 처리(Process)에서 이루어지는 일의 절차 · 내용을 문서화한 것임

3) 미니스팩의 작성 목적
- 자료흐름도의 보완
- 최하위도의 최하위 처리마다 작성해야 함. 시스템분석가를 위한 문서임

4) 미니스팩 작성도구
- 구조적 언어
- 디시전 테이블
- 순서도
- 디시전 트리
- N-S Chart
- 선후조건문(pre/post conditions)

* 구조적 언어
- 과거 분석 명세화의 문제점
    - 내용이 길고 복잡하여 이해가 어려웠음
    - 애매모호성으로 혼란, 또 다른 문제 유발
    - 기술방법 미표준화로 개인차가 심함
    - 용어 · 상태 · 상황기술이 현실과 유리되는 현상 발생함
- 과거 방법의 문제점에 대한 대안 중에 하나가 구조적 언어임
- 구조적언어
    - 제한된 단어, 제한된 문장 형태, 제한된 구조로 미니스팩(mini spec)을 작성 가능한 명세기술 언어(Specification Description Language)임

* 구조적 언어와 유사한 기능

㉠ 프로그램 기술언어 (Program Description Language : PDL)
㉡ 의사코드 (Pseudo Code)
㉢ 문제기술언어 (Problem Statement Language : PSL)
- 구조적 언어의 문장구조 표현 : 교재「PP. 92~94」참조
    - 순차(sequence) 구조
    - 선택(decision) 구조
    - 반복(iteration) 구조
- 구조적 언어의 사용상 유의사항은 다음과 같음
    - 이해 용이한 단어를 사용해야함
    - 수정 용이성 배려해야함
    - 논리적 구조의 표현에 ⇦Indentation을 사용함
    - 미표준화 되었기 때문에 융통성을 확보해야 함
- 구조적 언어의 사용상 유의사항은 다음과 같음
    - 영어로 표현할 경우
    → 자료사전에 나타난 이름, 중요 단어, 프로그램 블록 이름 대문자로 표기함
    → 기타 내용 소문자로 표기함

    - 다른 도구(tool)와 복합적으로 사용하면 효과적임
    - 순차구조, 선택구조, 반복구조로 명확하게 표현해야함
    - 논리표현에 ⇨「AND, OR, GT, GE, LE」등을 사용 가능함 : 교재「PP.94~95」참조

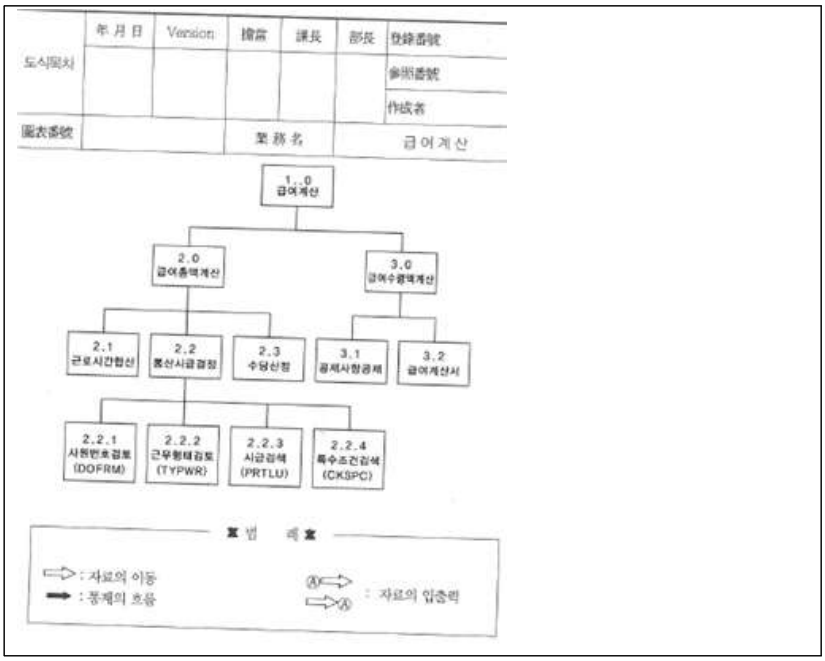
3. HIPO Chart

1) HIPO의 장점
- 하향식 개발이 용이함
- 문서화의 체계적으로 이루어짐
- 판독·이해의 용이성 제고
- 기능·데이터의 의존관계가 동시에 표현됨
- 보수·갱신·유지보수가 용이해짐
- 시스템 구상에 도움이 되며, 설계보조 수단으로 적합함

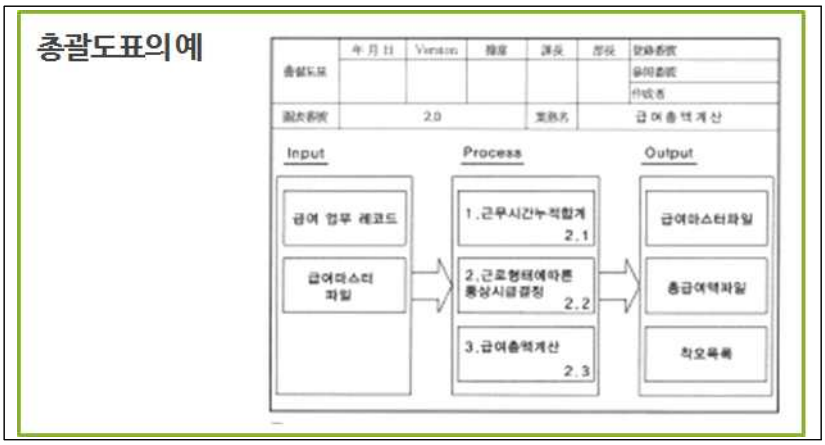
2) HIPO 도표
- 도식목차(VTOC : visual table of contents)
- 총괄도표(overview diagram)
- 상세도표(detail diagram)

3) 도식목차
① 계층구조도
- HIPO(Hierarchy plus Input-Process Output)에서 계층(H : Hierarchy)을 표현함

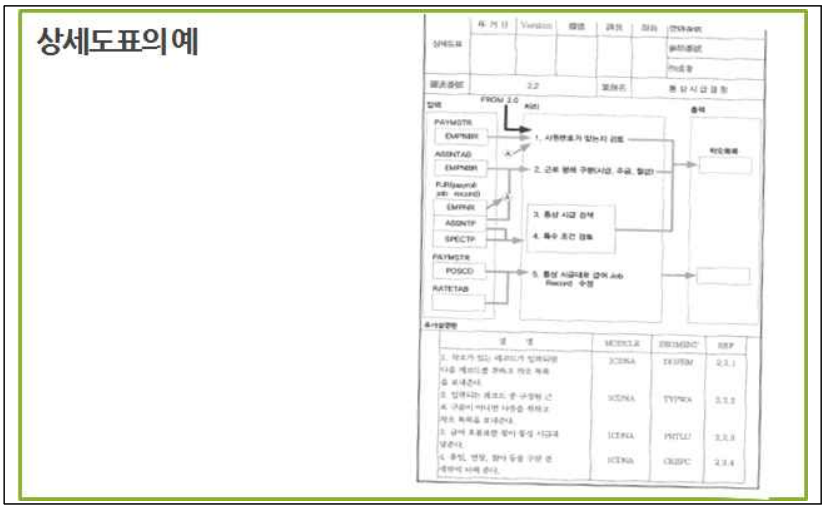
② VTOC
- 하위계층의 도표를 쉽게 찾도록 해줌
- 업무의 개괄적 이해를 돕는 문서임
- VTOC의 각 Box에 대해서 총괄도표, 상세도표가 작성됨
- VTOC의 예



4) 총괄도표
- 입력란, 처리란, 출력란 3 개의 부문으로 구성됨
- 각 란에 해당 내용을 직사각형 박스로 만들어 기입함
- 입력, 처리, 출력 부분을 화살표로 연결함
- 총괄도표의 예 : 교재「P. 98」의「그림 4-33」 참조



5) 상세도표
- 기본적인 구성은 총괄도표와 동일함
- 총괄도표와 차이점
  - 입력·처리·출력란에 기재한 내용을 화살표로 구체적인 관계를 표시함
  - 추가 설명란을 이용하여 ⇦보조설명이 가능함
- 상세도표의 예 : 교재「P. 100」의「그림 4-34」 참조



6) IPO 도표(Input Process Output diagram)
- 총괄도표 및 상세도표는 모두 공통적으로「입력란·처리란·출력란」으로 구성되어, 이들 두 가지 도표를 총칭하는 개념임

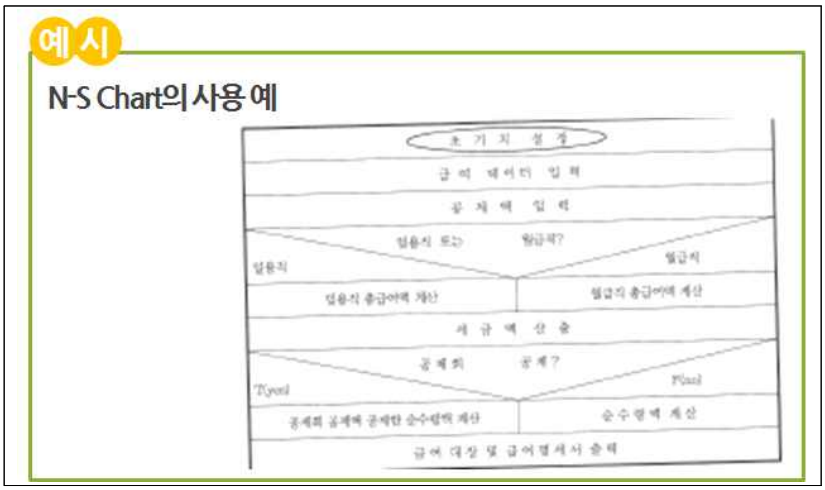
7) HIPO 문서의 단점은 다음과 같음
- 문서량이 증가함
- 작성에 많은 시간이 소요됨
- 과거 설계도구인 순서도(flowchart) 대신 사용이 불가능함

4. N-S Chart

- Nassi와 Shneiderman에 의해서 제안된 도표임
- 논리기술에 중점을 두는 그림형식으로 표현하는 도구(tool)임
- N-S Chart의 제어구조
  - 순차구조, 선택구조, 반복구조, 케이스 구조
- 사용 기호와 의미 및 사용 예 : 교재「PP. 102~104」의「그림 4-35」 참조

1) 장점
- N-S Chart로부터 직접 코딩(coding)이 가능함
- 테스트의 누락방지가 용이함
- 테스트케이스(test case) 설계 시에 테스트 경로(test path) 식별이 용이
- 교육·유지보수 문서로서 기능이 양호함
- 순서도 대신 사용이 가능함
- 구조적 코딩(structured coding)이 용이함

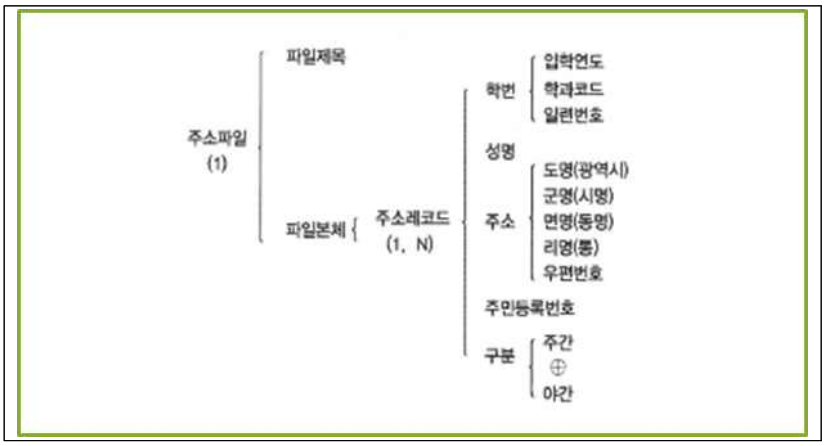
<N-S Chart의 사용 예 : 교재「P. 106」의「그림 4-36」 참조>



5. 워니어-오어 차트(Warnier-Orr Chart)

- Warnier-Orr 차트 : Warnier와 Orr 두 사람의 이름을 합성하여 만들어진 이름임
- 구조화 설계 도구
- Warnier-Orr 차트 : 시스템, 프로그램, 데이터 등의 계층적 구조를 도표로 표현하는 도구임
- HIPO 차트
- 구조도(structured chart)

1) Warnier-Orr 차트
- 수학의 집합이론을 근간으로 함
- 수학에서 집합의 구성요소 ⇦「가로」로 나열함
- 이 차트에서 집합의 구성요소 ⇦「세로」로 나열함
  - 단, 오른쪽 괄호는 사용하지 않음
- 사용 기호와 의미 : 교재「PP. 107~109」 참조



- 데이터의 표현 방법은 다음과 같음
  - 데이터 ⇨계층적구조로 표현 ⇦교재「P. 109」의 「그림 4-41」 참조
  - 판독법
    → 왼쪽에서 오른쪽으로 읽어감
    → 상단에서 하단으로 읽어감
  - 표기법 : 중괄호({ })를 중심으로
    → 왼쪽 : 기능이름을 기재함
    → 오른쪽 : 구성요소「세로」로 기재함

- 프로그램 구조의 표현은 다음과 같이 함
  - 순차구조, 선택구조, 반복구조를 표현 가능함. : 교재「P. 110」의「그림 4-42」 참조

  - 순차구조 : 「세로」로 차례대로 기입함
  - 반복구조 : 기능이름(집합명) 밑의 괄호 속에 반복횟수를 표기함- 선택구조
    → 기능이름(집합명) 밑에 (0,1)을 기입함
    → 실제 사용 예 : 교재「P. 110」의「그림 4-42」에서 「신규예약자처리」 참조
    → 제어논리 표시 : 해당 위치에 「」와 「숫자」를 기입함, 하단에 「」에 대한 설명을 기입함.

2) Warnier-Orr 차트의 장단점은 다음과 같음
- 계층적 구조만 기술하고, 데이터베이스에 기초를 두지 않음
- 도표의 본체에 제어논리가 표현되지 않아서 판독이 어려움
- 소규모이거나 단일 파일구조의 출력 중심의 문제의 설계, 문서화에 적합함
- 블록구조를 이용하면 구조화 프로그램 코드로 변환이 가능함
- 프로그램 구조 표현보다 데이터 구조를 표현하는 데 효과적임

6. 프로그램 기술 언어

1) 과거의 설계도구
- 스파게티 로직(Spaghetti Logic)이 발생하기 쉬움
- 논리의 누락·오류 발견이 어려움
- 논리 기술(Description)의 표준화 방안이 없었음

2) 프로그램 기술언어(Program Description Language : PDL)의 기능은 다음과 같이 요약됨
- 원래 구조적 프로그래밍 지원 도구임
- 최근 구조도(Structured Chart) 모듈의 내부 논리 기술에 사용하는 설계도구(문서화 도구도 됨)임
- 모듈의 기능 및 논리설계를 특정한 언어의 문법적 제약 없이 자연언어(Natural Language)로 표현 가능함

3) 자연언어를 사용함으로써 얻어지는 장점(특징)
- 시스템이나 프로그램을 서술적으로 기술함으로써 문서화로도 사용 가능함
- 프로그램의 기능과 순서 문법적 제한 없이 자연언어로 표현 가능함 그러므로 가상코드(pseudo code)임
- 프로그램 기술언어의 장점 : 교재「P. 112」 참조
- 고급요원에게는 불필요한 과정의 추가로 비효율적임

4) 프로그램 기술언어 작성법은 다음과 같음
- 순차구조 : 교재「P. 112」의「그림 4-43」 참조
- 선택구조 : 중첩된 조건을 표시할 경우 ⇦Indentation을 사용함. ⇦교재「P. 113」의「그림 4-44」 참조
- 반복구조 : 교재「P. 114」의「그림 4-45」 참조
- CASE 구조 : 교재「P. 114」의「그림 4-46」 참조

【학습정리】

1. 자료흐름도
- 작성자의 편의성 제고, 사용자의 이해의 용이성 제고 등을 위하여 단계화 한다.

2. 자료흐름도의 단계화
- 최상위도(context diagram)
- 중위도(middle diagram)
- 최하위도(배경도 : functional primitives)