Regular paper | Journal of the KIECS. pp. 403-408, vol. 19, no. 2, Apr. 30. 2024, t. 124, pISSN 1975-8170 | eISSN 2288-2189
http://dx.doi.org/10.13067/JKIECS.2024.19.2.403

구조적 방법론과 정보공학 방법론을 통합하는 프로그램 개발 과정 가이드와 사례

조민호*

Program Development Process Guide and Examples that Integrate Structural and Information Engineering Methodologies

Min-Ho Cho*

요 약

소프트웨어의 개발 방법은 환경의 변화에 따라 계속해서 발전되었다. 구조적 방법론, 정보공학 방법론 그리고 객체지향 방법론이 이런 변화의 큰 흐름이다. 각 방법론은 특징과 장점이 있고, 적용되는 분야가 다르다. 실무를 수행하는 엔지니어들은 각 방법론을 적용하는 것은 익숙하지만, 구조적 방법론이나 정보공학 방법론을 통합하여 적용하는 것에 익숙하지 않다. 그래서 방법론이 가지는 효과를 최대한 얻을 수 없다. 이런 점에 착안하여 각 방법론의 적용 대상에 대해 구조적 방법론, 정보공학 방법론을 통합하여 적용하는 방법에 대한 것을 정리하고 사례를 제시하고자 한다.

ABSTRACT

Software development methods have continued to evolve in response to changes in the environment. Structural methodology, information engineering methodology, and object-oriented methodology are the major trends of this change. Each methodology has its own characteristics and advantages, and its applied fields are different. Engineers performing practical work are familiar with applying each methodology, but are not familiar with applying structural methodologies or information engineering methodologies in an integrated manner. Therefore, the maximum effect of the methodology cannot be obtained. With this in mind, we would like to organize and present examples of how to integrate and apply structural methodology and information engineering methodology to the target of each methodology.

키워드

Structured Methodology, Information Engineering Methodology, Software Development, Integrated
구조적 방법론, 정보공학 방법론, 소프트웨어 개발, 통합

I. 방법론의 발전

프로그램의 개발은 프로그램 언어를 사용하여 수행되게 된다. 프로그램 언어를 분석하면, 제공하는 기능은 “변수의 선언”, “변수의 타입”, “순환과 반복”의 3가지 기능에 “함수”가 추가된 것을 확인할 수 있다.

* 교신저자 : 중원대학교 컴퓨터공학과
• 접 수 일 : 2024. 01. 08
• 수정완료일 : 2024. 02. 24
• 게재확정일 : 2024. 04. 12
• Received : Jan. 08, 2024, Revised : Feb. 24, 2024, Accepted : Feb. 12, 2024
• Corresponding Author : Min-Ho Cho
Dept. Computer System Engineering, JungWon University,
Email : chominhokr@jwu.ac.kr

방법론은 프로그램을 구성하는 5가지 요소들을 잘 활용하고자 고안된 것이다. 전체적인 방법론의 발전을 정리하면 그림 1과 같다.

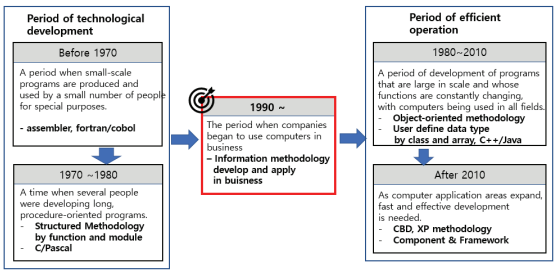

그림 1. 방법론의 발전 역사
Fig. 1 History of methodology

그림 1의 내용을 정리하면, 프로그램은 처음에는 어셈블러를 이용하는 순차적인 처리에서 시작하였고, 순차적으로 처리해야 하는 프로그램의 길이가 길어지면서 프로그램의 가독성이 떨어지게 되었다. 이것을 보완하기 위하여 함수라는 기능을 프로그램에 추가하게 된다. 이때, 함수를 이용하여 프로그램을 제작하기 위한 업무의 분석과 처리에 대한 과정을 체계적으로 정리한 것이 구조적 방법론이다.

이후 데이터베이스가 도입되면서 기업에서 프로그램을 사용하기 시작하였는데 기업에서 프로그램을 개발하는 데 필요한 것을 체계적으로 정리한 것이 정보공학 방법론이다.

프로그램이 일반 기업과 국가 기관을 포함한 모든 업무에 사용되면서 프로그램의 길이가 길어지고, 여러 개발자가 프로그램 개발에 참여하게 되고, 프로그램의 기능이 지속해서 추가, 변경되는 상황이 발생하게 되었다. 이런 환경에 적합한 프로그램을 개발하기 위해 개발된 것이 클래스라는 자료형을 기반으로 프로그램을 개발하는 객체지향 프로그램이며, 이것을 개발하기 위한 과정을 체계적으로 정리한 것이 객체지향 방법론이다.

추가로 프로그램을 개발할 때, 개발자가 공통 기능을 공유하면서 개발하면 효과적으로 개발할 수 있음을 확인하고, 개발자의 공통 기능 모듈(이것을 컴포넌트라고 한다)을 이용한 프로그램 기법이 도입되어 현재 사용되고 있다.

요약하면, 프로그램의 개발에 관련된 방법론은 총 3가지가 개발되어 사용되고 있다. 이중 구조적 방법론과 객체지향 방법론은 프로그램 개발을 위한 환경 및 구성의 설계에 대한 것이다. 객체지향 방법론과 컴포넌트를 이용하는 것은 프로그램 자체의 개발에 대한 것이다. 이번 연구의 대상은 프로그램 제작에 필요한 환경 분석을 효과적으로 수행하기 위하여 관련된 구조적 방법론과 정보공학 방법론은 통합하여 적용하는 과정과 사례를 보여 주는 것이다. 그래서, 이후에 다루는 내용에서는 객체지향 방법론과 컴포넌트를 활용하는 것에 대해서는 다루지 않는다 [1-3].

II. 구조적 방법론의 적용 방향

구조적 방법론은 프로그램의 제작 기술 중에서 함수를 이용한 프로그램의 제작을 체계적으로 수행하도록 만들어진 것이다[4-7], [10-11].

업무용 프로그램을 제작할 때, 함수를 사용하기 위해서는 함수로 표현할 단위 업무를 식별하는 것이 필요하다. 이것을 모듈이라고 한다. 그리고 모듈은 입력과 출력 그리고 모듈 내부에서 수행해야 하는 작업에 관한 서술이 필요하다. 모듈을 이용하여 프로그램 수행 과정을 모형화하는 것을 DFD(Data Flow Diagram)라 한다. DFD의 사례가 그림 2에 있다.

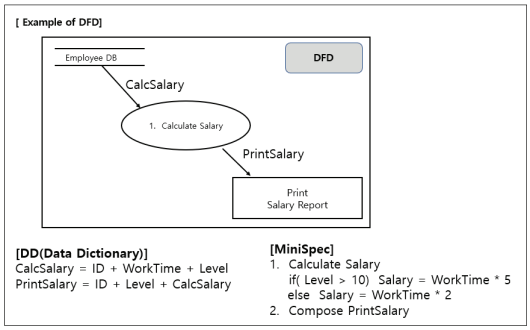

그림 2. DFD의 예
Fig. 2 Example of DFD

DFD를 그리기 위해서는 입력과 출력에 사용되는 데이터에 대한 정의가 필요하다. 이때 사용되는 것이 DD이다. 그림 2에서는 Calculate Salary 모듈에 대한 입력을 CalcSalary가 정의 되었고, CalcSalary는 DD

에 ID+WorkTime+Level로 정의되어 있다.

Calculate Salary 모듈에서 수행하는 기능에 대한 것도 정의해야 하는데, 이때 사용하는 것이 MiniSpec이다. 그림 2에서 보면 MiniSpec에 Salary 계산 과정이 구체적으로 설명되어 있다.

구조적 방법론은 DFD, DD, MiniSpec으로 구성된다. 프로그램해야 하는 업무를 모듈 단위로 나누어서 DFD로 표현하고, DFD의 입출력 데이터를 정의하고, DFD의 내부 수행 절차를 명시함으로써 프로그램을 위한 준비를 마치게 된다.

III. 정보공학 방법론의 적용 방향

정보공학 방법론은 기업에서 컴퓨터를 적용하는 것을 목적으로 한다. 이를 위하여 가장 중요한 것은 어떤 데이터를 다룰 것인지를 정하는 것이다. 기본적인 정보공학의 구성은 3 sides, 4 levels이다. 그림 3에 이에 대하여 요약하였다[8-11].

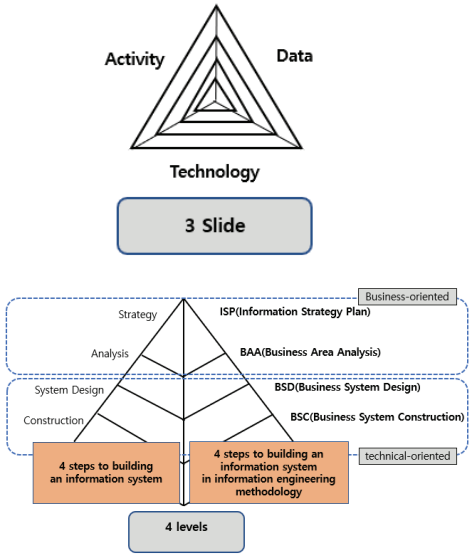

그림 3. 3 sides, 4 levels in 정보공학 방법론
Fig. 3 3 sides, 4 levels in IE Methodology

그림 3에서 3 sides의 의미는 기업에서 컴퓨터를 적용하려면, 데이터(Data)와 처리 절차(Activity) 그리고 기술(Technology)가 고려되어야 한다는 의미이다. 이를 구현하기 위한 단계는 ISP, BAA, BSD 그리고 BSC의 4 단계(4 levels)로 구성된다.

정보공학의 기본 개념을 파악하였으므로, 정보공학에서 가장 중요한 과정은 ISP임을 이해했을 것이다. ISP(전략정보계획)의 단계를 정리한 것이 그림 4에 있다.

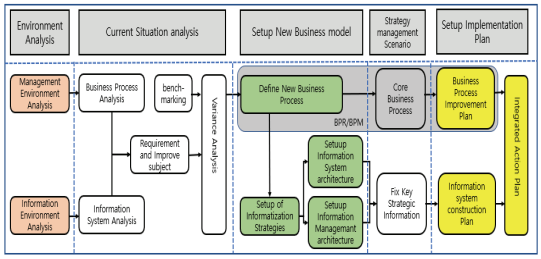

그림 4. 정보공학 방법론 중 전략정보계획 절차
Fig. 4 ISP procedure in IE Methodology

전략정보계획의 의미는 기업에서 컴퓨터를 적용하기 위해서는 기업의 경영환경과 정보환경을 동시에 고려해야 한다는 점을 강조한 것이다. 경영환경분석에서 비즈니스 프로세스를 알아내고, 정보시스템과 연계하여 요구 및 개선 사항을 파악한 다음, 차이분석을 수행한 이후, 컴퓨터와 통합된 새로운 비즈니스 프로세스를 정의한다. 새롭게 정의된 비즈니스와 정보시스템의 구축 계획이 동시에 고려되어 통합실행 계획이 수립되는 것이다.

요약하면, 현 업무처리 절차와 정보시스템 환경을 분석해서 요구사항을 정하고, 다른 회사를 분석하여 보완한 다음 정보시스템 환경을 고려한 새로운 비즈니스 프로세스를 정의하고 실행 계획을 수립하는 것이다. ISP가 끝난 후에 BAA 과정을 수행하게 되는데, 이때 수행하는 일 중에서 가장 중요한 것이 데이터베이스 관리시스템(DBMS)의 설계이다. 이때 쓰이는 기법이 개체-관계 모델링(E-R Modeling)이다.

최종적으로 기업에서 사용하는 데이터베이스를 설계하는 것과 이를 이용하는 업무처리 절차를 새롭게 설계하는 것이 정보공학 방법론의 목표가 된다. 앞에서 설명한 내용을 전체적으로 정리하면 그림 5와 같이 표현할 수 있다.

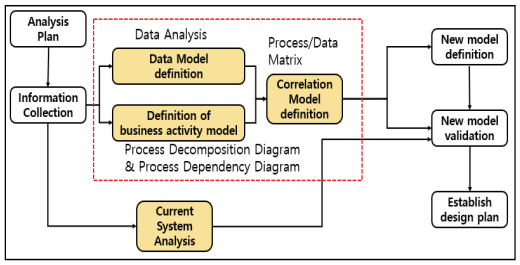

그림 5. 정보공학 방법론의 핵심 과정 요약
Fig. 5 Core process summary of IE Methodology

IV. 통합 적용을 위한 가이드

구조적 방법론과 정보공학 방법론은 각자 목표가 명확하게 구분되는 방법론이다. 구조적 방법론은 개발해야 하는 프로그램을 기능적인 관점에서 모듈로 구분하고 이들이 어떤 순서로 수행되어야 하는지에 관하여 관심을 가지고 개발된 것이다.

객체지향 방법론은 개발해야 하는 프로그램이 다루어야 하는 데이터의 구성에 중심을 두고 개발된 것이다. 그러므로 두 개의 방법론은 우리가 개발해야 하는 프로그램에서 고려해야 하는 프로세스와 데이터를 위한 것이므로 두 방법론을 조화있게 사용하는 것은 우리가 개발해야 하는 프로그램의 품질과 직결된다고 할 수 있다.

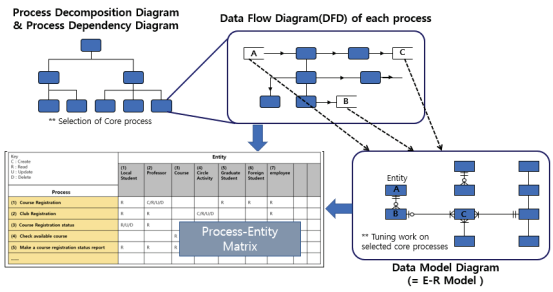

그림 6. 정보공학 방법론과 구조적 방법론의 조화
Fig. 6 Harmony of Structured and IE methodology

그림 6은 구조적 방법론과 정보공학 방법론의 조화로운 사용을 위한 개념을 그림으로 표현한 것이다. 먼저, 개발해야 하는 업무처리 절차를 Top-Down으로 분석한 다음, 각 업무처리 절차별로 단위 기능인 모듈을 식별한다. 식별된 모듈을 대상으로 업무처리 절차별로 Process Dependency Diagram을 그린다. 이 과정이 처음 수행되어야 하는 과정이다.

이 과정이 완료된 후에는 Process Dependency Diagram의 모듈별로 Process Decomposition Diagram을 생성한다. 여기에는 각 모듈을 구성하는 단위 작업(업무의 복잡도에 따라 보다 작은 모듈) 이 처리 순서대로 표현되게 된다.

개발해야 하는 업무가 간단하다면 Process Decomposition Diagram을 한 단계로 종료할 수도 있지만, 업무가 복잡하다면 2~3단계로 진행될 수도 있다. 중요한 점은 Process Decomposition Diagram을 구성하는 모듈은 더 이상 분리하면 의미가 없는 최소한의 단위이어야 한다는 점이다. 지금까지의 과정을 쉽게 이해할 수 있도록 실무에서 적용된 사례를 정리한 것이 그림 7이다. 제시된 사례는 주문을 받고, 재고를 확인하고, 제작, 조합 및 창고입고의 과정을 정리한 것이다.

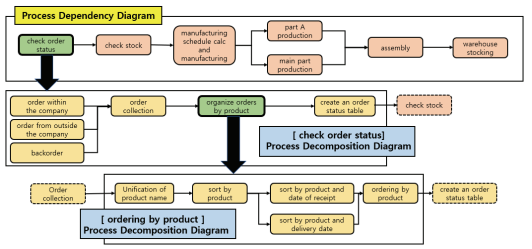

그림 7. 프로세스 분석 다이어그램 간의 관계
Fig. 7 Relationship between Process Analysis Diagram

그림 7의 윗부분은 개발해야 하는 특정 업무처리 과정에 대하여 큰 의미의 모듈을 정의하고 이들 간의 연관 또는 종속성을 그림으로 표현하였다(Process Dependency Diagram).

이를 통하여 개발해야 업무에 대한 전체적인 파악이 되었다면, 이제는 모듈의 내부를 살펴볼 시간이다. 이때 작성하는 것이 모듈별로 작성하는 Process Decomposition Diagram이다. 이를 구성하는 모듈은 일반적으로 단위 기능 수준이 될 것이다. 만약 업무가 복잡하여 단위 기능으로 표현하지 못했다면 한 단계 더 분석할 수 있다. 그림 7은 Process Decomposition Diagram을 두 단계 분석한 사례를 제시하고 있다.

앞의 과정을 통하여 우리가 개발해야 하는 업무에 대한 프로세스들과 각 프로세스를 구성하는 모듈을 파

악하였고, 각 모듈을 단위 작업의 수준이 될 때까지 분석하였다면, 이제 각 모듈에 대하여 구조적 방법론에서 제공하는 DFD를 그릴 차례이다. 그림 6에서 최종 모듈별로 DFD를 그리는 것을 예로 보여주고 있다.

모듈별로 구성된 DFD를 통합하는 과정을 통하여 최종적인 DFD를 얻을 수 있게 된다.

최종 DFD를 구성하는 데이터저장소를 대상으로 개체 관계 모델링을 적용하여 데이터베이스 디자인을 수행할 수 있다. 이때, 정보공학 방법론에서 제공하는 데이터베이스 디자인 기법이 적용된다.

앞의 과정을 수행한 후에는 반드시 수행해야 하는 것은 아니지만, 앞에서 발굴한 업무 프로세스와 통합 DFD의 데이터저장소를 기반으로 모델링 개체(Entity) 간의 연관성을 확인하는 과정이 필요하다. 이 부분은 그림 6에서 확인할 수 있으며, 구체적인 모습은 그림 8에 있다.

Key C : Create R : Read U : Update D : Delete	Entity								
Process	(1) Local Student	(2) Professor	(3) Course	(4) Circle Activity	(5) Graduate Student	(6) Foreign Student	(7) employee		
(1) Course Registration	R	C/R/U/D			R	R	R		
(2) Club Registration	R	R		C/R/U/D			R		
(3) Course Registration status	R/U/D	R			R/U/D	R/U/D	R		
(4) Check available course			R						
(5) Make a course registration status report	R	R	R						
.....									

그림 8. 프로세스 개체 매트릭스
Fig. 8 Process-Entity Matrix

그림 8은 그림 7에서 선정한 프로세스와 개체(entity)간의 관계를 CRUD를 통해 서술하는 것이다. 이것을 통하여 대규모 업무 분석을 수행하다 보면 생략되거나 중복되는 프로세스나 개체를 확인할 수 있다. 추가로 여러 프로세스 간에 통합이 가능한 상황의 확인도 가능하고, 프로그램 전체에 대한 디자인을 최적화할 수 있다.

앞에서 설명한 절차를 통하여 DFD, 개체-관계 모델링 그리고 프로세스 개체 매트릭스가 유기적으로 연결되어 프로그램의 완성도를 높이는 데 이바지한다.

V. 결론 및 향후 연구 방향

이번 연구는 대형 프로그램의 설계를 위하여 사용하는 구조적 방법론과 객체지향 방법론의 기법을 조화롭게 적용하기 위한 절차를 개발하는 것에 중점을 두고 진행하였다.

두 방법론이 각자 다른 팀에 의해 적용되는 현실의 상황에서 두 방법론은 유기적으로 연결하여 설계할 수 있다면 보다 양질의 프로그램 개발이 가능할 것이다.

이를 위하여 프로그램의 설계를 Process Dependency Diagram으로 시작하였다. 이 과정을 통해서 전체 프로그램을 구성하는 업무처리 절차들을 파악하고, 각 업무처리 절차를 구성하는 모듈 간의 연관성을 파악하는 것이 목적이다.

이후에는 프로세스를 구성하는 각 모듈을 단위 작업의 수준까지 낮추기 위하여 모듈별로 Process Decomposition Diagram을 작성한다. 이는 개발해야 하는 업무를 구성하는 기본 모듈을 식별하기 위한 과정이다.

이렇게 식별된 모듈들은 프로세스별로 존재하게 되고, 이것을 통합하여 최종 기본 모듈을 식별한다.

이 과정이 완료된 후에 모듈별로 DFD를 작성하는 과정을 진행하게 된다. 수많은 DFD를 구성한 후에 이들에게서 데이터저장소에 대한 정보를 정리할 수 있게 되고, 이것을 기반으로 개체-관계 모델링을 수행함으로써 개발해야 하는 업무에 맞는 데이터베이스를 설계할 수 있다.

이번 연구는 별도로 진행되는 방법론을 Process Dependency Diagram의 개념을 기반으로 통합하여 수행하는 방안에 대하여 개념과 절차 그리고 실제 적용하여 개발한 사례를 제시하였다. 이번 연구가 대규모 프로그램을 제작하는 현장에서 유용하게 사용되어 개발되는 프로그램의 품질 향상에 이바지하기를 바란다.

향후 이번 연구에서 자세하게 설명하지 못한 모듈을 구성하는 DFD의 통합에 관한 부분을 보완하여 실무자에게 도움이 되는 기법을 제시할 예정이다.

References

[1] H. B. Jun and H. W. Suh, “A QCD Analysis Framework Based on the System

Development Methodology of Information Engineering," *Entrue Journal of Information Technology,* vol. 8, issue. 2, July 2009, pp. 31-40.

[2] M. Mukherjee, "Object-Oriented Analysis and Design," I*nternational Journal of advanced engineering and management,* vol. 1, issue 1, Dec. 2016, pp. 18-24.

[3] H. J. Bae, "Comparative Analysis on the Attributes of NHPP Software Development Cost Model Applying Gamma Family Distribution," *J. of the Korea Institute of Electronic Communication Sciences,* vol. 18, no. 5, oct. 2023, pp. 867-876.

[4] A. Mishra and Y. L. Alzoubi, "Structured software development versus agile software development : a comparative analysis," *International Journal of System Assurance Engineering and Managemtnt,* vol. 14, Jan. 2023, pp. 1504-1522.

[5] M. H. Cho, "A Study on the Direction of Modeling Techniques for the Development of Large Scale Software," *J. of the Korea Institute of Electronic Communication Sciences,* vol. 15, no. 1, Jan. 2020, pp. 167-172.

[6] C. H. Choi, C. B. Shim, S. W. Park and S. H. Jung, "Implementation of a Mobile App for Companion Dog Training using AR and Hand Tracking," *J. of the Korea Institute of Electronic Communication Sciences,* vol. 18, no. 5, oct, 2023, pp. 927-934.

[7] S. Ghaemmaghami and L. Bucciarelli, "Structured Mathods in Product Development," *Internation Jounal Engng Ed,* vol. 19, no. 1, 2003, pp. 132-141.

[8] S. Y. Jang, "Designing an Intelligent Data Coding Curriculum for Non-Software Majors: Centered on the EZMKER Kit as an Educational Resource," *J. of the Korea Institute of Electronic Communication Sciences.* vol. 18, no. 5, oct, 2023, pp 901-910.

[9] B. Chung and S. Yun, "The Procedure for Improving Structural Methodology or Information engineering Methodology," *Journal of Korea Information Processing Society,* vol. 9, no. 6, 2002, pp. 108369-1090.

[10] B. Song and Y. Yu, "A Design and Implementation of Software Architecture for IPC in Vehicles Using Modeling Methodology," *J. of the Korea Institute of Electronic Communication Sciences*, vol. 7, no. 6, Dec. 2012, pp. 1321-1326.

[11] J. Choi and C. Choi, "Analysis on the current status of the fourth industrial revolution-oriented curriculum of the computer and software-related majors based on the standard classification," *J. of the Korea Institute of Electronic Communication Sciences,* vol. 15, no. 3, Jan. 2020, pp. 587-592.

저자 소개

조민호(Min-Ho Cho)

1989년 인하대학교 졸업(공학사)
1989년 ~ 2012년
HP Korea, Openwave, SK C&C에서 개발자, 컨설팅, PM의 역할을 수행

2003년 숭실대학교 대학원 컴퓨터공학과 졸업 (공학박사)
2013년~ 중원대학교 컴퓨터공학과 교수
※ 관심분야 : 인공지능 및 기계학습, 데이터분석, 소프트웨어공학, 데이터 마이닝 및 빅데이터