연구논문집 제 10권 제 2호 p.17~19 1979

Computer Software 開發에 있어서의 効率的인 工程管理에 關한 연구

高 在 鎭
電子計算學科

〈要 約〉

Software 開發에 있어서의 工程과 각 區分工程에서의 管理方法. 그리고 결과로 나오는 生産物의 區分과 管理方法에 대해서 논했다.

A Study on the Effective Process Management of the Computer Software Development

Ko, Jae-Jin
Department of Computer Science

〈**Abstract**〉

This Paper Describes the Processes of the Software Development, the Management Methods of the Each Partitioned Process, the Classification of the Products as the Results of the Development, and the Management Methods of the Products.

Ⅰ. 서 론

Computer가 가동되기 위해서는 필수적으로 Software가 필요하게 된다. Software란 Computer를 구성하는데 있어서 기계적인 本體인 Hardware를 제외한 전기능을 말한다.

예를 들면, Computer를 제어하고, 부과된 작업을 수행하는 Operating System, Data를 보존, 수정, 作成, 管理 하는 Data 管理 Utility, Assembler Language를 기계어로 번역해서 실행가능한 Program으로 만드는 Assembler, High Level Language를 번역하는 Compiler 등… 무수히 많은 Software가 필요하게 된다.

이러한 Software를 개발하는데 있어서의 工程管理上의 문제점에 대해서 시술하려고 한다.

Computer에 있어서의 Software의 역할이란 막중한 것이어서 Computer의 性能, 處理能力, 活用度를 높이는데 決定的인 것이다.

Ⅱ. Software 開發의 過程

Software는 여러過程을 거쳐서 개발되는 것이다. 첫째로 Software는 Computer 使用者의 일반적인 필요성과 요구에 의해서, 보다 System을 효율있게 운용하기 위해서, 사용빈도가 많고, 공통적인 기능을 발휘하는 Software의 필요성에 의해서 開發이 착수되는 것이다. 그리고, 그 Software의 장래성, 상업적가치, 학문적가치등 여러 요건을 감안해서 개발착수가 결정되면, 자료조사와 Research에 들어가게 된다.

자료 조사를 하는데 있어서의 要點은 우선 類似 Software를 조사하고, 그 長短點을 연구 검토하고, 보다 향상된 Software를 만들기 위한 기초 조사를 하게 된다. 그리고, 다른 Software와의 互換性, 그 Software가 動作할 Hardware, Software 的인 환경을 면밀히 검토하고, Level-up의 전망, 他機種과의 Interface등을 조사해서 기본설계에 반영하게 된다.

上記와 같은 인구, 조사 단계가 끝나면, 그 Software의 基本的인 仕樣, 性能, 動作 환경을 設計하는 基本設計에 들어가게 된다. 이 기본 설계는 주로, 전문가라든지, 개발경험이 많은 베테란이 참석하게 되며, 開發의 기본방향이 여기서 결정되게 된다.

기본 설계가 끝나면 生産物로서 基本設計書가 나오고, 그 설계서를 기초로해서, 그 Software를 구성하는 各 Module의 기능을 설계하는 機能設計에 들어가게 된다. 기능설계에서는 各 Module의 기능 구성, 他 Module과의 Interface, 共通情報의 참조관계, 外部仕樣의 결정, Error 處理方法의 決定등 구체적인 기능을 설계한다.

기능설계가 끝나면 생산물로서 기능설계서, 구성사양서, Interface 사양서가 나오게 된다. 그 다음은, 기능설계를 기초로 해서 詳細設計에 들어가게 된다. 상세 설계에서는 Program의 Logic을 설계하고, 變數, 領域을 決定하고, Coding의 기초가 되는 사항을 決定하게 된다. 상세설계가 끝나면 Coding에 들어가게 되고 Coding이 완료되면, 作成된 Program을 Assembler나 Compiler로 Compile을 해서 目的 Module을 만들게 된다. 目的 Module이 완성되면 Linkage Editor를 이용해서 Load Module (實行可能한 Program)을 만들게 된다.

Load Module이 完成되면 各 Module 單位의 Test인 單體 Test를 한다. 各 Module의 기능이 정상적으로 수행되는가를 Test하는 單體 Test가 끝나면 Module 結合 Test에 들어 간다. 이 結合 Test는 Module間의 Interface가 제대로 이루어지는 지를 Test 하는 것이다. 結合 Test가 끝나면, Software의 全體的인 기능을 Test 하는 總合 Test를 한다. 총합 Test가 끝나면, 최종적으로 성능, 품질 Test를 해서 合格하면 User에 提供된다.

User에 제공된 후에도 障害가 발생하면 保守(Maintenance)를 해 주어야 한다.

이와 같이 Software의 開發은 복잡한 여러 단계를 거쳐서 행해지는 것이다.

Ⅲ. Software 開發의 諸生産物

Software의 本體는 Computer에 Install 되어서 機能을 발휘하고 있는 Program을 말한다.

Software 開發의 生産物로서 Program 자체만을 말한다면 단순한 처사이다. 동작되고 있는 Program의 保守, Level-up, 他 Software의 開發등을 위해서 여러가지의 Documentation이 있어야 한다.

Software 開發의 諸生産物을 열거하면 다음과 같다.

a. 調査報告書
b. 基本設計書
c. 機能設計書
d. 構成仕樣書
e. Interface 仕樣書
f. 詳細設計書
g. Source List
h. 試驗仕樣書
i. Test Program
j. 試驗結果報告書
h. 外部仕樣書
l. Install Tape
m. 使用手引書
n. System 編集手引書
o. Debugging Tool
p. 保守手引書

Ⅳ. Software 開發의 各 工程別 管理方法

1. Research 및 調査

이 工程에서는 Software 전문가 및 開發 담당자가 광범위한 자료조사 및 연구를 하게된다. 開發할 Software의 성능, 處理能力, 互換性, 外部仕樣, User의 희망, 使用빈도 등을 광범위하게 조사해서 Research 한다. 결국, 開發의 必要性을 규명하고, 開發方向을 提示하는 것이 이 공정의 역할이다.

2. Basic Design(基本設計)

이 工程에서는 開發 Group의 核心 Member가 참가해서 開發할 Software의 基本機能, 基本構成, 外部仕樣, 實行論理, 開發思想 등을 設計하게 된다. 그리고, 性能과 處理能力의 基準値를 設定하고 다른 Software와의 互換事項을 規定하게 된다.

定期的으로 Group Seminar를 열어서 設計의 共通事項을 討議해서 決定하고, 이미 설계된 사항을 再檢討(Review)해서 未備한 사항이 있으면 修正한다.

즉, 設計 -Review- 修正의 과정을 거쳐서 設計

를 진행시킨다.

3. Function Design(機能設計)

이 工程에서는 各 Module 別 담당자가 決定되고 外部仕様의 各項目別 機能에 대한 구체적인 내부처리 방법을 決定하고 Module의 기능을 구체화 한다. 그리고, Module의 構成을 決定하고, 他 Module과의 Interface를 맞춘다. Design 方法은 設計 -Review- 修正의 과정을 거친다. 定期的으로 Group Review會를 열어서 設計된 내용을 검토, 토의하게 된다.

4. Detail Design(詳細設計)

이 공정에서는 기능설계서를 기초로해서 Program Logic을 設計한다. Program Logic을 설계하는 방법에는 여러 가지가 있는데, 一般的으로 Flow Chart를 많이 사용한다.

근래에 와서는 Pseudo Code 方式이라든지 HIPO 方式등이 많이 사용된다. No Go To 型 Program을 開發하자는 움직임이 일고 있는데 이것은 기능단위당 한개의 Subroutine을 할당하는 것이다. Go To 文을 많이 쓰면 Program을 이해하는 데나 Debug 하는데 어려움이 많기 때문에 Pseudo Code를 써서 Detail Design 하는 方法도 좋다.

5. Making(Ⅰ)

이 공정에서는 상세설계서를 기초로해서 Program을 Coding 해서 Compile 해서 目的 Module을 만든다. Coding을 할 때 要點이 되는 것은 Memory 領域을 적게 사용하고, 實行速度가 빠르게, 또, Program을 이해하기 쉽게끔 Comment를 상세히 기입하는게 좋다.

6. Making(Ⅱ)

이 工程에서는 작성된 Program을 Module 別로 Test하는 것이다. Test 하기전에 미리 Test 仕様書를 작성해서 Test 方法, Test 項目을 決定한다. 그리고, Test 仕様書에 준해서 Test Program을 作成해서 Test에 들어간다.

7. Making(Ⅲ)

이 공정에서는 各 Module을 결합해서 Test 한다 여기서 Test하는 것은 Module間의 Interface의 確認에 있다.

8. Component Test

이 공정에서는 Software의 全기능을 Test 한다. Test 순서는 먼저 기본기능을 Test하고, 그 다음 확장기능, 추가기능을 Test 한다.

9. 性能 Test

이 공정에서는 Software를 User에 提供하기전에 최종적으로 성능과품질을 Test하게 된다.

기준에 합격한 Software 만이 User에 제공된다.

10. Maintenance

User에 제공된 Software는 제공된 이후에도 障害발생에 대해서 Maintenance를 하게 된다.

Ⅴ. 結 論

Software 開發에 있어서의 効率的인 工程管理는 各 區分工程別 엄격한 공정 Check와 各 工程別生産物의 철저한 保守, 管理에 있다. 各 工程別 評價會를 열어서, 지난공정의 못다한 사항과 문제점을 반성하고, 수정을 한 다음에 그 다음의 공정으로 넘어가야 한다. 設計과정에서는 Review會를 열어서 여러 Member로부터 다각적인 의견을 경청해야 한다.

참 고 문 헌

1. Software 開發技法, 富士通(株), 1975.
2. Barry W. Boehm, "Software Engineering", IEEE Trans. on Computers, Dec. 1976, pp. 1226—1241.