한국산학기술학회논문지
Vol. 8, No. 6, pp. 1428-1442, 2007

웹 환경 시스템 구축을 위한 레거시 마이그레이션 개발방법론

이준웅[1*], 양해술[1]

Legacy Migration Development Methodology for Construction of Web environment system

Jun-Woong Lee[1*] and Hae-Sool Yang[1]

요 약 최근 IT환경의 급 변화로 인하여 하드웨어, 소프트웨어와 소프트웨어 개발방법론 등 다양한 기존 인프라 시스템이 급변하는 추세에 있다. 레거시시스템은 그 동안 IT산업이 발전해 온 수많은 과정을 통해 투자한 예산, 시간 등을 비롯 기업의 핵심적 지적 재산이 대량 포함되어 있다. 한국뿐만 아니라 일본, 유럽, 미국 등은 레거시시스템 처리에 어려움을 겪고 있으며, 레거시시스템을 재사용하거나 이를 처리하는 방안을 위한 개발방법론과 이를 해결하는 지원도구 구현에 대한 깊이 있는 연구가 진행 되어야 할 시점이다. 본 논문에서는 레거시시스템에 대한 정의와 개념을 나아가 실질적 시장 추세와 공급사, 그리고 기술적 트렌드를 통해 등장 배경을 밝히고, 레거시시스템을 처리하는 구체적인 개발 방법론 절차를 제안하며, 적용사례와 기대효과를 분석하였다. 레거시 마이그레이션을 통해 레거시 시스템의 재사용 및 유지보수 비용 절감으로 수익성의 증대와 생산성 향상을 얻을 수 있었다.

Abstract Nowadays, as IT environment is changing rapidly, traditional infra system such as hardware, software, and development methodology are changing as well. Legacy system has been filled with Intellectual property such as invested budget and development period for the time of IT is developing. Many countries around the world are having troubles with how to manage the legacy system. It is time to study on a development method and the implementation of supporting tool to reuse or process the legacy system. In this paper, it proves the definition, concept and also background of legacy system by showing real market and technology trend. Developed to handle the concrete development methodology, procedure and reference. We can get more profitability and productivity advance by reuse of legacy system and maintenance cost-saving through legacy migration.

Key Words : Legacy System, Legacy Migration, 소프트웨어 재사용, 프로그램 변환

1. 서 론

기업의 성공여부는 e-비즈니스 모델을 어떻게 구축하느냐에 따라 그 성패가 결정되는 상황이 되었다[1]. 기업의 경영은 정보화시스템을 효율적으로 적용하느냐에 따라 기업의 성패가 결정된다[2]. 1960년대 말에 이르러 미국의 NATO를 중심으로 소프트웨어공학 기법이 적용되어야 한다는 원칙 아래 체계적인 작업을 시작하였고 프로그램 언어, 데이터베이스 등이 하드웨어와 접목되면서 메인프레임 시대를 여는데 기여하였다. 그 이후 많은 하드웨어 산업이 기하급수적인 발전을 하면서 클라이언트/서버 시스템을 통하여 다운사이징이 시작되었지만 소프트웨어의 기본적인 골격은 기존 방식을 그대로 유지하고 있었다. 그 이후 폭발적인 인터넷 열풍에 따라 대부분의 기업들이 e-비즈니스를 도입하게 되었다.

또한, 세기말 재앙인 Millenium Bug(이하 Y2K)는 대부분의 기업들이 정보기술 분야에 투자하는 것을 위축시키는 요인으로 작용하였지만, Y2K 문제가 원활히 해결되어 대부분의 기업들이 정보기술 분야에 대규모 투자를 진행하고 있다[3]. 기업은 e-비즈니스와 맞지 않는 메인프레임 시스템이나 폐쇄적인 환경의 시스템에 투자하는 것을 억제하고 새로운 오픈 환경 시스템을 가지게 되었

본 연구는 정보통신부 및 정보통신연구진흥원의 대학IT 연구센터 지원사업의 연구 결과로 수행되었음.
(IITA-2007(C1090-0701-0032))
[1]호서대학교 벤처전문대학원 정보경영학과
*교신저자: 이준웅(lee_junw@hanmail.net)

다. 그러나 기존의 폐쇄적인 환경에서 운영되는 레거시시스템은 반드시 해결하고 넘어가야할 부분으로 그 중요성이 강조되고 있다.

기업의 신정보시스템 도입에는 세 가지 방법이 있는데, 새로운 관점에서 업무를 분석하고, 설계, 구현하여 적용단계까지 개발하는 것과 패키지화되어 있는 전사적자원관리시스템(Enterprise Resource Planning, 이하 ERP)을 도입하는 것, 그리고 레거시시스템을 활용하여 기존에 사용했던 소프트웨어를 재구성하여 사용하는 방법이 있다. 이 중에서 레거시시스템을 재활용하는 방법이 지적재산 승계, 비용절감, 자원활용, 개발기간 단축 그리고 신규개발에 따른 위험을 줄이는 방법으로서 가장 이상적이라고 할 수 있다.

레거시시스템은 나라별 사회적 환경에 따라 사용하는 형태가 다르다. 국내에서도 소프트웨어 시스템의 목적에 따라 시스템의 생성과 소멸이 매우 급격하게 이루어져 왔다. 따라서 많은 시행착오, 과도한 투자, 개발 시간의 증가 등 비효율적인 생산체계로 전개되었다. ERP기술이나, 새로운 Web-enable시스템 도입으로 기업의 중요한 지적재산을 지닌 레거시시스템을 폐기하는 과오를 초래하고 있다[1]. 또한 많은 IT 예산 집행과 신규개발에 대한 위험을 감수하면서 신규 프로젝트를 개발함으로써 많은 문제점을 내포하고 있는 실정이다. 이와는 반대로 일본, 미국, 유럽 등에서는 안정된 사회적인 시스템 기반 아래 체계적으로 개발이 진행되어 왔다. 이는 개발자들이 레거시시스템을 중요한 가치를 지닌 자산으로 판단하고 있기 때문이다. 정보기술이 발달된 나라들은 레거시시스템에서 비즈니스-규칙(Business-rule)을 추출하여 새로운 오픈 시스템으로 전개하는 방식인 마이그레이션 체제로 전환할 필요성을 절실히 느끼고 있다. 본 연구에서는 기업의 많은 지적재산과 업무 프로세스를 담고 있는 레거시시스템의 철저한 분석과 이를 기반으로 새로운 프로그램 환경으로 마이그레이션하기 위한 개발방법론을 제안하고 사례 적용에 관하여 진행하였다.

첫째, 레거시시스템의 분석경험과 국외에서 활발하게 진행하고 있는 소프트웨어 재사용(Software reuse)에 관한 연구결과들을 분석하였다.

둘째, 웹 환경 시스템 구축 개발을 위한 개발방법론(RE-Method(v2.0))을 설계하여 개발하였다.

셋째, 개발방법론을 적용한 사례와 기대효과를 기술하였다.

본 논문의 2장에서는 상기 내용에서 언급한 레거시시스템에 대한 관련연구를 기술하였으며, 3장에서는 Web 환경 시스템 구축개발을 위한 레거시시스템의 마이그레이션 개발절차 및 방법론에 대해 제안하였으며, 4장은 정리된 개발방법론에 의한 실제 적용사례와 기대효과를 중심으로 전개하였다.

2. 관련 연구

2.1 레거시시스템의 등장

레거시란 단어의 IT용어 개념으로는 "현재의 기술 보다 이전의 프로그램 언어와 플랫폼 기법으로 만들어진 어플리케이션과 데이터"라고 정의되어 있다. 또 온라인 용어 전문 사이트인 www.webopedia.com 에서는 "레거시 시스템(또는 레거시어플리케이션)은 낡은 기술과 하드웨어 및 플랫폼 기반에서 운영되는 어플리케이션을 말하며 이를 새로운 플랫폼 즉, 웹 환경의 Java, .NET기반의 C# 환경으로 전환하는 것을 레거시시스템을 마이그레이션한다." 라고 정의하고 있다. 본 연구에서는 레거시시스템의 범위를 단순히 오래된 프로그램 언어나 하드웨어로 제한하지 않고, 새로운 신기술에 적합한 개발방법론, 플랫폼, 프로그램 언어, 데이터베이스 등을 포괄하는 시스템이라

표 1. 레거시시스템의 현황

구 분	레거시 현황	
하드웨어	AS/400, System36-38, Data General, Ticom, VAX 등	
프로그램	3세대언어	COBOL, C, PL/I, FORTRAN, Assembly, Pro*c, Store Procedure 등
	비쥬얼언어	Delphi, Visual Basic, Power Builder, VC++등
	4세대언어	Progress, Centura, SQLWindows, Ideal, Unifact, Object view, Forte, Natural 등
데이터베이스	Flat file, Index file(ISAM, VSAM, KSAM), Datacom, Adabas, IMS/DB 등	
미들웨어	CICS, IMS/DC, Tuxeudo, Entera, Topend, Tmax 등	
Screen	DDS(AS/400), Screen file, Screen Section, BMS(CICS)	
공급업체	Ticom, Data General, Gupta 등	
개발방법론	전통적 개발방법론, 자체적으로 작성된 방법론, 정보공학방법론	

는 전제로 진행하였다.

2.1.1 레거시시스템의 등장배경

레거시시스템의 탄생은 하드웨어의 급격한 발전과 이를 구현하는 소프트웨어의 구조 변화에서 찾을 수 있다. 새로운 패러다임이 등장하게 되면 하드웨어는 교체되지만 소프트웨어 교체는 그리 쉽지 않다는 점이 레거시시스템 탄생 배경이다. [표 1]은 레거시시스템의 현황에 대한 내용이며 단순히 프로그램 언어뿐만 아니라 하드웨어, 프로그램, 데이터베이스, 미들웨어까지 레거시시스템을 운영하기 위한 내용을 포함하였다.

아래 [표 2]는 레거시 마이그레이션을 위한 목표시스템 현황이다.

표 2. 레거시시스템의 목표 시스템 현황

구 분	현 황
프로그램	Microsoft's C#.net VB.net, EJB for J2EE
데이터베이스	Oracle, DB2, 기타 표준 RDBMS
WAS	Websphere, WebLogic, ZEUS
개발방법론	CBD(Component Based Development), RUP, Select, Fusion, MaRMI-III 등

기술적인 측면에서 레거시시스템의 탄생은 프로그램 언어의 구조적인 변화과정에서 찾아 볼 수 있다. 초기의 독립적인 프로그램 구조에서 구조적 프로그램 구조로 변화하고 다시 클라이언트/서버 구조로 변화되어 현재는 웹 분산처리 시스템구조로 변화하고 있다. 즉, 인터넷 환경에서 컴포넌트 기반의 소프트웨어로 제작이 되어 재사용이 가능한 형태를 취하고 있으므로, 재사용을 위한 가장 이상적인 프로그램 구조라 할 수 있다.

레거시시스템이 등장한 것은 단순히 공급자 및 기술적 현상만은 아니며 사용자의 환경적인 측면에서 그 배경을 찾을 수 있다.

① 유지보수 비용의 증대

eWEEK 발표에 따르면, 순수한 메인프레임 환경에서 운영되는 프로그램언어에 사용된 소스코드 라인은 2,000억 라인이며 이를 개발한 비용도 US $20조에 육박한다고 발표하고 있으며, 매년 20%씩 소스코드가 새롭게 작성되고 있다는 것은 레거시시스템에 엄청난 개발 자금과 시간 및 유지보수 비용이 요구된다.

② 유지보수의 어려움

업무 프로세스와 기술의 급 변화로, 이를 지원하기 위한 레거시시스템의 유지보수 문제는 신속, 정확, 적시에 대응해야하는 어려움에 직면하게 되었다.

③ 정보기술의 급변

정보기술은 빠르게 변화하고 있으므로 레거시시스템 처리문제도 IT부서에서 보면 기술적으로 어렵다고 판단되기 때문에 많은 비용과 시간을 투자해서 변화를 시도하고 있다.

2.2 리엔지니어링의 일반적 모델

소프트웨어 리엔지니어링의 시작은 레거시시스템의 소스코드로 시작되며 다섯 단계를 거쳐 목표시스템의 소스코드를 생성하는 것으로 결론을 맺는다. 이러한 프로세스는 단순히 소스코드를 다른 소스코드로 변환하는 것은 물론, 하나의 플랫폼을 다른 플랫폼으로 전환하는 포괄적인 의미를 가지고 있다.

반면에 소프트웨어 리엔지니어링은 매우 복잡하다. 현재 소스코드를 사용하여 재설계하거나 현행 시스템에 들어 있는 내용과 새롭게 요구하는 내용을 비교 검토하거나 시스템을 컴포넌트기반 설계로 재구조 혹은 재설계하는 시스템의 형태를 가지로 최종적으로는 목표(To-Be) 환경의 소스코드를 생성해내야 하는 복잡한 과정을 거치게 된다. [그림 1]은 소프트웨어 리엔지니어링의 개발 프로세스이며, 중요기능과 As-Is시스템에서 역공학적인 프로세스에서 시작하여, 순공학적 접근을 통한 To-Be시스템을 설계 및 구현하도록 구성되었다.

① 추상화(Abstraction): [그림 1]의 좌측에서 보는바와 같이 As-Is시스템의 개발방식은 역방향의 개념으로 발전되어 간다.

② 변경(Alteration): 추상화의 정도를 변경하지 않고 시스템에 하나 이상의 변화를 표현하는 것이다.

③ 구체화(Refinement): [그림 2]의 우측 To-Be시스템의 추상화개념에서 구체화 해나가는 과정을 의미한다.

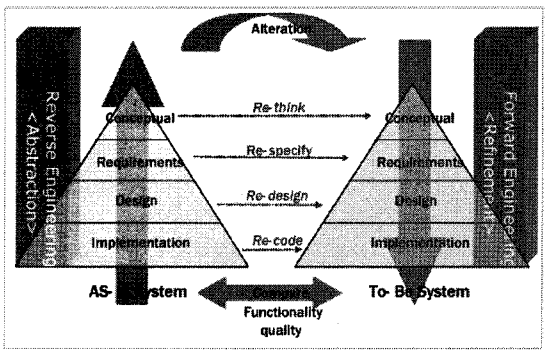


그림 1. 소프트웨어 리엔지니어링의 모델

2.3 레거시 마이그레이션(LM)의 필요성

레거시시스템을 컴포넌트 기반의 환경으로 전환하는 것은, 컴포넌트 기반의 환경이 정확성, 확장성, 재사용성이 뛰어나며, 소프트웨어의 품질에 절대적인 영향을 미치기 때문이다. 또한 글로벌 표준화가 컴포넌트 기반으로 발전하고 있는 추세는 더욱 레거시시스템이 컴포넌트 기반으로 가야 하는 필연성을 갖게 한다[9].

2.3.1 재사용의 중요성과 절차

레거시시스템을 신규로 구축하는데 드는 비용은 초기 투자비용의 50%이며, 개발 기간도 2-3년 정도 걸리기 때문에 기업들은 많은 부담을 가지고 있다. 만일 기업에서 보유하고 있는 지적재산인 레거시시스템을 표준 플랫폼으로 마이그레이션하는 전략과 개발방법론을 활용한다면 많은 문제점을 해결할 수 있게 된다. LM의 궁극적인 목적은 새로운 정보기술 환경으로 전환하는 것이며 필요한 기술은 데이터베이스, 사용자 인터페이스, 그리고 프로그램 로직을 개발하는 기술로 압축된다. IT 예산의 불필요한 지출, 간이기술로 중복 투자, 레거시시스템 기술자의 활용 문제 등은 LM의 중요성을 더욱 부각시키고 있다. 레거시시스템의 복잡성을 단순하게 처리하는 것은 어렵기 때문에 LM을 착수하기 전에 다음의 절차를 수행해야 한다.

- 레거시시스템의 이해: 고립된 레거시시스템의 기능을 분석하여 이해도를 극대화하는 절차를 실행한다.
- 레거시시스템의 분석(마이닝): 현행시스템에 최소한의 수정을 가하여 데이터 혹은 레거시시스템을 확장하거나 마이닝을 실행한다.
- 레거시시스템의 전환: 레거시시스템을 새로운 언어나 기술 환경으로 래핑(Wrapping) 변환하고 Re-Architect 기술을 사용한다.

2.3.2 컴포넌트 기반의 소프트웨어

CBD 접근은 소프트웨어를 컴포넌트화하여 재사용하게 하는 방법으로, 현재 소프트웨어를 재사용하기 위한 중요한 개발 패러다임으로 자리매김하고 있다. 레거시시스템에서 컴포넌트를 추출하여 재사용하는 장점은 다음과 같다.

첫째, 복잡한 문제를 작고 간단하게 나눌 수 있어(컴포넌트, 작은 단위로 분해) 초기개발 보다 실질적으로 수행할 수 있게 해 준다.

둘째, 컴포넌트는 소프트웨어를 캡슐화하였기 때문에 오류가 발생하여도 캡슐의 범위 내에서 영향을 미치므로 오류의 영향을 최소화할 수 있을 뿐만 아니라 유지보수가 용이하다.

셋째, 어플리케이션을 컴퓨팅 네트워크에 분산 배치될 수 있는 단위로 볼 수 있다.

넷째, 어플리케이션이 동일한 인터페이스를 통하여 코드와 데이터를 공유할 수 있게 한다.

3. LM을 위한 개발방법론

3.1 LM 개발방법론 이론적 배경

LM 개발방법론은 기존의 방법론인 마르미-RE(V1.0)에 실제 적용사례가 부족하고, 개발절차를 업그레이드하지 못한 채 마무리가 되었다. 또한 실제로 레거시 마이그레이션의 다양한 사례에 있어서 적용이 쉽지 않았다. 단순히 마르미-RE(v1.0)의 절차와 기법 등을 2003년 이후 줄곧 연구 및 적용과정을 거치면서 소프트웨어 LM의 이론적인 근거를 소프트웨어 리엔지니어링 분야에서 찾아보기로 하고, 웹 환경 시스템 구축을 위한 레거시 마이그레이션 개발방법론을 연구하였다. LM 개발방법론과 지원도구의 이론적 기반이 되는 소프트웨어 리엔지니어링의 프로세스를 살펴보면, [그림 2]와 같이 목록분석, 문서 재구성, 역공학, 코드 재구성, 데이터 재구성, 그리고 순공학 등으로 구성되어 있다. 다시 말해, 리엔지니어링 프로세스에 맞게 개발방법론의 절차를 구축하고 이를 지원하는 형태로 분석하고 구현하였다.

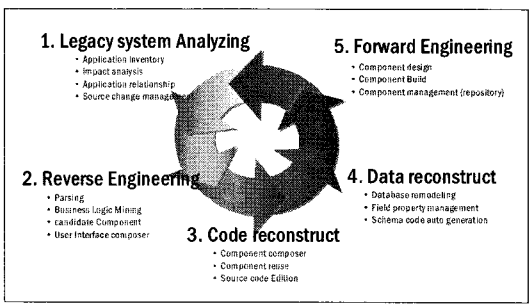


그림 2. 소프트웨어 리엔지니어링의 프로세스

3.1.1 레거시시스템을 활용한 역공학 목표 모델

소프트웨어 리엔지니어링 프로세스는 역공학을 통해 잃어버린 추상화 즉, 설계, 분석 요구사항과 같은 정보를 발견함으로 "horseshoe" 모델이 탄생하게 되었다[6]. 그 후 순공학을 통해 하위단계인 실행 단계로 점차적으로 이동하는 방식을 말한다. 가장 하위 단계는 레거시 코드가 있으며, 가장 상위단계는 아키텍처(Architecture)가 차지하고 있다. 역공학 프로세스는 아키텍처를 복구하는 기능을 수행하고 있으며, 순공학의 프로세스는 아키텍처 기반의 개발에 초점을 맞추고 있다. 본 연구는 이러한 이론적 배경으로 개발방법

론을 설계하였으며, 개발방법론은 구조화 및 비구조화된 레거시 소스코드를 어떻게 목표모델에 도달시키는가를 기술한다.

[그림 3]은 리엔지니어링 모델을 기반으로 어떻게 레거시시스템에서 요구하는 최종 모델을 구현하는가를 나타낸 것이다. (1)은 프로그램 안에 산재되어 있는 프로그래머의 주석부분과 조각화된 기술 기반위에 만들어진 알고리즘을 추출하는 과정을 설명한다[7]. (2)는 만일 레거시 코드가 구조적 프로그램으로 작성되어 있다면, 바로 구조화된 프로그램 단계로 이동한다. 만일 비구조화된 프로그램 코드라면 구조화된 프로그램 형태로 변환한다. (3)은 비구조화된 프로그램 코드를 추출하여 Go To문을 최소화한 상태로 조정하여야 한다. (4)는 비구조화된 프로그램 코드를 정제(refine)한 다음 Go To문을 최소화 하도록 소스코드를 재조정한다. (5)는 구조화된 프로그램코드를 파싱(분석)작업을 통하여 추상화 구문구조(AST, Abstract Syntax Tree)를 주석이 달린 그래픽 형태로 나타낸다. (6)은 주석이 달린 그래픽 구조를 목적에 부합되는 형태의 그래픽 형태로 다시 나타내는 것을 말한다. (7)은 검증이 가능하도록 비기능 요구사항을 정의한다. 끝으로 (8)은 품질 메트릭스 기법과 같은 기능을 활용하여 비기능 요구사항을 추출한다.

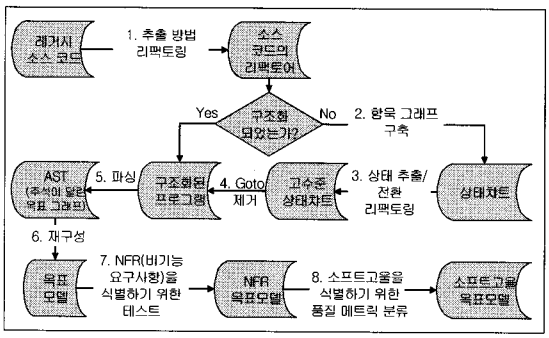


그림 3. 소스코드 재구조화 프로세스

3.2 LM 개발방법론의 설계

3.2.1 개발방법론의 비교

LM 개발방법론은 "마르미-RE(레거시시스템 컴포넌트화 방법론)" 버전을 기반으로 이를 보완하여 LM 개발방법론을 실제 현업에 적용하였다[14].

(1) 대상 프로그램 언어의 제한성

마르미-RE는 기존의 COBOL과 같은 비표준화 구조를 갖는 언어를 중심으로 새로운 플랫폼 환경인 Java 환경으로, EJB 매핑 전략을 중심으로 극히 제한적인 작성으로 실제로 다양한 플랫폼을 갖는 시장에서 이를 적용하기에는 많은 어려움으로 인해 절대적으로 보완이 필요하였다.

(2) 업무 적용 범위 부분에서 제한적

마르미-RE는 '레거시시스템을 컴포넌트화 하는 개발방법론'이라고 하는 반면에 보완된 개발방법론은 레거시시스템을 RE-5 (Re-Documet, Re-Face, Re-Place, Re-Hosting, Re-Architect) 전략을 제공함으로서, 사용자의 환경에 맞는 마이그레션 프로세스를 지원한다.

- Re-Document: Re-Document는 역공학을 이용한 분석을 통하여, 현행시스템의 자산정보, 영향분석, 연관관계 등의 정보를 추출한다.
- Re-Face: Re-Face는 레거시어플리케이션 부분에서 UI 부분만을 추출하여 Target System으로 변환/보완하는 방법이다.
- Re-Place: Re-Place 방식은 비즈니스로직을 그대로 자동변환하는 방법으로, 기존의 언어를 새로운 Target 언어로 변환한다.
- Re-Hosting: Re-Hosting은 현행시스템을 오픈시스템으로 수평이동하는 것이며 다운사이징 개념을 도입한 Mainframe을 Unix Open System으로 이동하는 방법이다.
- Re-Architect: Legacy System을 분석하고 재사용 컴포넌트를 자동 추출하여 이를 보완 및 교체하여 새로운 아키텍쳐의 시스템을 생성하는 방법이다.

(3) 자동화된 도구의 지원 여부

마르미-RE는 개발 계획 단계에서 우선 프로세스를 제작한 후에 도구를 개발하기로 하였지만 이를 지원하지 못하고 있다. 그러나 보완된 개발방법론은 프로세스도 중요하지만, 지원도구와 연결을 중심으로 개발되었기 때문에 많은 효율성을 기대할 수 있도록 하였다.

(4) 레거시시스템에 대한 분석 기능의 강화

마르미-RE는 단순하게 컴포넌트를 추출하기 위한 정도의 레거시시스템을 분석하였지만, 사용자의 요구는 자신의 환경과 요구(혹은 LM전략)에 맞게 도입한다. 보완된 개발방법론인 RE-Method는 레거시시스템을 단순한 분석정보에서 영향분석, 변경관리, 중요도 관리(Function point), 시스템 자원간의 인터페이스 등 다양한 기능을 제공하는 자동화된 문서를 제공하고 있다. 이를 기초로 다양한 마이그레이션 요구에 중요한 기초 자료로 활용 및 마이그레이션 이후에도 지속적인 관리의 기능을 제공하게 된다.

(5) 개발방법론의 실용화 부분에서 많은 차이점

마르미-RE는 초기의 연구결과이었지만, RE-Method는

실제 프로젝트에 적용하면서 이를 보완하고 기능을 대폭 강화하여 현장에 적용함으로써 그 성과를 구체적으로 나타낸 점이다.

3.2.2 개발방법론의 단계와 절차

RE-Method(v2.0)은 레거시시스템을 컴포넌트 기반의 웹 환경으로 전환하기 위한 개발방법론이다. 또한 향후 재공학적인 프로세스를 통하여 가장 최신 기술인 레거시 시스템을 Java나 C#과 같은 컴포넌트 기반의 플랫폼으로 마이그레이션하는 것을 목적으로 삼고 있다. 또한 LM을 이행하지 않더라도 자신들이 운영하고 있는 As-Is시스템을 역공학단계를 거친 후, 다양한 As-Is 시스템의 구조, 문제점, 연관관계, 데이터 혹은 프로그램이 시스템에 미치는 영향분석 등의 관리문서를 자동으로 분석한다. 이하에 제안된 RE-Method의 계획단계, 역공학단계, 컴포넌트화단계 등 주요 3단계의 기능을 정리하였다.

(1) 계획단계(Planning phase)

레거시시스템의 유지보수 및 운영에 관련된 전반적인 정보를 분석한다. 목표시스템에 대한 비즈니스 유스케이스 모델링을 하여 개선된 비즈니스모델을 제시하고, 이를 바탕으로 프로젝트의 목적과 범위를 결정한다. 각 업무 단위별로 컴포넌트화를 위한 변환전략을 수립하고 수립된 변환전략에 따라 개발 프로세스를 정의하여, 최종적으로 개발계획서를 작성한다.

① 레거시시스템의 현황파악: 업무의 전반적이고 개괄적인 정보분석을 통해 조직의 구조와 업무의 흐름, 조직이 당면한 최대 이슈를 파악하고, 업무의 기능과 단위 업무별 서브시스템의 기능을 이해한다.

② 개선 비즈니스 도출단계: 비즈니스 유스케이스 모델링과 비즈니스 객체모델링을 통해 이해 당사자의 요구사항을 파악하여 향후 목표로 하는 개선 비즈니스 모델을 제시하며, 이를 바탕으로 프로젝트의 목적과 범위를 결정한다.

③ 개선전략 수립단계: 개선 대상 업무를 선정하고, 이에 대한 비즈니스 가치 및 시스템 측면의 기술적 요소들을 분석하여 재공학의 우선순위를 결정하며, 각 업무 단위별로 컴포넌트화를 위한 최적의 변환전략을 수립한다.

④ 개발계획수립 단계: 결정된 컴포넌트 변환 전략에 따라 개발 프로세스를 수립함으로써 개발에 실제 적용할 방법론의 절차와 산출물에 대한 항목들을 선택하며, 이전 작업의 산출물들을 취합 및 정리하여 개발계획서를 작성한다.

(2) 역공학 단계(Reverse Engineering phase)

재공학의 대상이 되는 레거시시스템에 대한 산출물들을 수집하고, 원시코드를 중심으로 분석작업을 수행하며, 참조할 수 있는 레거시시스템의 도메인 정보를 분석한다. 분석한 정보를 바탕으로, 설계정보의 모델링 작업을 통해 레거시시스템의 정적, 동적, 행위 정보를 이해한다. 레거시시스템의 구성 요소들 간의 관계 인식을 통해 아키텍쳐 정보를 이해하고 추상화시킴으로써, 컴포넌트화를 위한 준비 작업을 수행한다. 레거시시스템을 유지보수하고 새로운 환경으로 전환하고자할 때 시스템에 대한 이해 증진을 목적으로 수행한다.

① 프로그램 분석(파싱): 레거시시스템의 산출물 및 참조할 수 있는 도메인 정보를 수집하여, 레거시시스템에 대한 인벤토리를 구축함으로써, 재공학을 위한 준비 작업들을 수행한다. 또한 문법적 파싱과 의미론적 파싱을 통해 어플리케이션의 정보를 정확하게 분류하게 된다.

- 어플리케이션 수집: 어플리케이션 즉, 프로그램, 데이터베이스 그리고 각종 라이브러리 등의 정보를 사용자가 제공하는 디렉토리나 매체를 통하여 수집하는 과정
- 어플리케이션 파싱: 수집된 어플리케이션 자원을 분석하는 과정을 수행하는데 있어, 해당언어가 지니고 있는 문법적 요소, 의미론적 요소 등을 감안하여 분해 작업 및 분석작업을 실시한다.
- 어플리케이션 인벤토리: 레거시시스템의 주요 구성요소로서 데이터와 응용시스템 외에 이를 지원하는 시스템 자원들이 있다. 즉, 레거시시스템의 기술 아키텍쳐의 현황을 분석하여 조직의 정보화 수준이 어느 정도인지를 분석하고, 운영환경, 개발환경, 테스트환경 등을 파악한다.

② 어플리케이션 분석: 수집된 레거시어플리케이션의 인벤토리를 세밀하게 분석하여 누락된 부분이나, 불일치하는 부분을 파악하여 이를 사용자와 함께 보완작업을 실시한다.

- 의미정보 등록: 레거시어플리케이션에 의미를 부여하기 위한 사전 작업으로 사용자의 관점에서 정보를 제공하고 이를 레거시어플리케이션과 연결작업을 실시한다.
- 프로그램 분석 정보: 레거시어플리케이션이 갖고 있는 프로그램 차원의 정보를 관리하는 단계로, 프로그램 구성과 이를 활용하기 위한 각종 서브장치 즉, 변수, 파라그래프를 포함한 정보를 제공한다.
- 시스템 구조 분석: 프로그램, 데이터베이스, JCL 등 어플리케이션의 각 자원 및 연관 정보를 심도 있게 분석

하는 단계이다.

- 업무(기능)분석정보: 역공학 기법을 활용하여 레거시 어플리케이션에서 표준용어 정보를 연결하여 다양한 정보를 제공하는데, 가장 중요한 업무 정보를 제공함으로 업무에 대한 심도 있는 정보를 파악할 수 있다.

③ 데이터 분석: 레거시어플리케이션에서 사용하고 있는 데이터정보를 분석하고 향후 목표시스템의 데이터베이스로 전환 또는 관리를 위하여 다양한 연관관계 등을 파악하기 위한 작업이다.

- 데이터 마이닝: 데이터 정보 이해는 레거시시스템을 구성하고 있는 주요 데이터 구조에 대한 정보를 추출함으로써 레거시시스템의 정적인 구조에 대한 보다 효율적인 이해를 돕기 위한 작업이다.
- ERD: 역공학을 이용하여 레거시시스템의 데이터구조를 ERD로 추출한다.
- 데이터베이스 분석정보: 레거시시스템의 데이터에 대한 정제화 및 정규화 작업을 실시한 후, To-Be 데이터로 전환한다.

④ 프로그램 재구조화: 프로그램 재구조화(Restructuring) 작업은 COBOL, PL/I과 같은 레거시 어플리케이션에 해당하며 프로그램의 불필요한 부분을 제거하고, 사용자의 의도대로 작성된 소스코드를 주어진 코딩 규칙에 의거하여 재 정돈하고 소스코드의 표준화에 따라 수정하며, 원문과 비교작업을 통해 구조화된 프로그램으로 작성하는 기능을 수행한다.

- 코드제거: 미사용 코드. 파라메터, 변수 및 주석을 제거한다.
- 코드정제: 코딩 규칙을 등록하여, 코딩 규칙에 위배되는 프로그램 소스를 발견하여 수정한다.
- 연관분석: 재구조화된 레거시 프로그램을 재포맷팅하며, 재구조화된 부분을 찾아 회귀시험을 반복한다.
- 재구조화비교: 시험데이터를 선정하고, 이를 입력시킴으로써 기능적인 변화유무를 파악한다. 또한 소스코드 및 플로우차트, 스트럭쳐 차트 등을 통하여 프로그램 소스의 변화를 확인한다.

⑤ 후보 컴포넌트 마이닝: 재구조화된 프로그램에서 재사용이 가능한 부분을 자동 혹은 수작업으로 추출하여 이를 정제하고 기존의 표준 컴포넌트와 비교하여 신규 컴포넌트일 때 이를 등록하는 작업을 수행하는 기능이다.

- 후보 컴포넌트 추출: 레거시시스템으로부터 역공학 과정을 통해 추출된 프로그램 분석 정보 및 실제 정보를 이용하여 독립된 비즈니스 기능을 제공하는 컴포넌트 후보들을 선정하고, 각 컴포넌트 후보들에 대해 그들을 구성하는 시스템 요소들을 프로그램 분석 정보를 이용하여 추적하고, 파악하는 과정이다.

[그림 4]를 보면, 프로그램의 종류에 따라 화면처리, 로직처리, 출력처리 등으로 구분함을 알 수 있는데, 출력 프로그램은 자동화툴(OZ제품)의 기본정보를 제공하는 절차를 거치며, 로직처리 프로그램은 자동변환 절차를 거쳐 이를 하나의 후보 컴포넌트로 저장하게 된다.

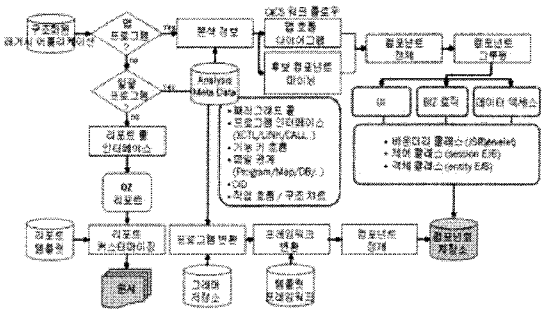


그림 4. 후보 컴포넌트 추출 프로세스

단지 화면 처리 프로그램은 세 가지 클래스 즉, 경계, 데이터처리, 로직처리로 분류하여 이를 기존의 컴포넌트와 비교하여 새로운 기능의 컴포넌트라고 판단이 되면 이를 재구조화-컴포넌트 그루핑 작업 등을 거치면서 후보 컴포넌트로 바뀌게 된다.

- 후보 컴포넌트 정제: 컴포넌트 파악 과정의 결과를 바탕으로 컴포넌트를 구성하는 시스템 요소를 인식한다.
- 후보 컴포넌트 등록: 표준 컴포넌트 저장소를 매핑작업을 통해 후보 컴포넌트를 정식으로 등록한다.
- 후보 컴포넌트 분석정보 제공: 컴포넌트에 대한 상세정보 및 현황을 제공한다.

(3) 컴포넌트화 단계(Componentization phase)

레거시시스템을 새로운 아키텍처의 목표시스템으로 이전, 변환, 컴포넌트로 전환하고 이를 조립하여 어플리케이션 프로그램을 개발하는 단계이다. 레거시시스템을 재공학하기 위한 방법과 이를 수행하기 위한 전략을 결정하고, 추출된 재사용 요소들의 컴포넌트화를 위해 소프트웨어 및 컴포넌트 시스템 아키텍처를 정의한다.

변환된 각 컴포넌트를 통합하여 시스템을 구축하는 과정에서, 관련된 컴포넌트들 사이의 인터페이스가 시스템 설계 과정에서 정의된 대로 통신 하는지, 통합된 시스템이 레거시시스템의 기능을 정확하게 구현하는지 여부를 시험한다. 어플리케이션 프로그램은 "데이터베이스", "사용자 인터페

이스", "비즈니스 로직"으로 구성되어 있으며, 변환의 결과물로는 "컴포넌트"로 집약될 수 있다.

컴포넌트로 저장된 형태는 비즈니스 로직 형태의 컴포넌트와 사용자 인터페이스의 장표 혹은 스크린에 관련된 패턴 컴포넌트로 구성되어 있는데 이를 조립(Composing)하여 최종적인 사용자가 요구하는 .NET프레임워크나 Java플랫폼 형태의 어플리케이션을 만들어주는 단계이다.

① 데이터베이스 변환: 역공학 단계에서 추출되고 정제가 된 스키마정보를 자동으로 생성하고 이를 기반으로 레거시데이터베이스의 정보를 목표의 데이터베이스(관계형 데이타베이스) 형태로 자동으로 변환하기 위한 절차를 통해 실질적인 자료까지 새로운 구조로 전환하는 과정을 거치게 된다.

- 스키마 자동생성: 데이터 마이닝 과정을 통해 얻은 정보를 ERD 솔루션으로 전환을 하여 새로운 데이터베이스 카다로그 정보 레파지토리에 저장한 정보를 기반으로 RDBMS의 스키마를 자동으로 생성하는 과정을 수행한다.
- 데이터베이스 변환: 변환을 위한 자동 매핑프로그램을 기본으로 As-Is파일/DB로부터 Flat파일의 형태로 변환작업을 실시한다. 그리고 변환된 정보의 검증작업을 거쳐 As-Is데이터를 To-Be데이터로 변환한다.

② UI 변환: UI 부분은 레포트와 스크린(화면) 정보를 의미하는데 보고서는 별도의 제품인 OZ솔루션으로 본 절차에서 자료를 제공하는 형태로 진행이 되며, 화면정보는 화면정보를 마이닝 작업을 통해 캡처한 다음에 이를 새로운 목표 화면 패턴에 맞게 변환하는 과정을 거치게 된다.

- 스크린 마이닝: 레거시시스템의 화면프로그램에 존재하는 화면정보를 추출한다. 다양한 형태의 추출된 화면정보를 취합 등 서로간의 연결관계 등을 모으는 작업을 진행한다.
- 스크린 변환: 레거시 화면을 솔루션의 화면편집기에 나타낸 후에, 추출한 화면정보를 입력함과 함께, To-Be환경의 화면으로 변환함과 함께, 사용자가 이를 수정 및 보완작업을 진행한다.

③ 비즈니스 로직 변환: 레거시시스템의 화면-프로세스 부분, 배치-처리 부분을 후보 컴포넌트로 추출하여 이를 저장한 저장소를 기반으로 목표가 되는 플랫폼(Java, .NET)에 맞게 자동으로 변환하는 프로세스 과정이다.

- 컴포넌트 매핑: 레거시시스템으로부터 추출한 후보 컴포넌트와 기존에 가지고 있는 컴포넌트와 기능비교하면서 중복된 후보컴포넌트를 제거하고 새로운 컴포넌트를 등록한다. 컴포넌트 매핑 프로세스는 [그림 5]와 같이 구성될 수 있다.

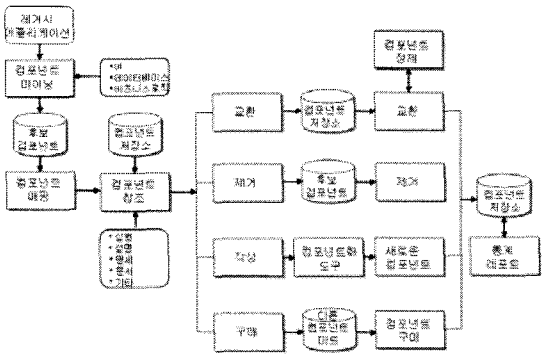


그림 5. 컴포넌트 매핑 프로세스

- 컴포넌트 변환: 매핑된 컴포넌트를 To-Be시스템 환경에 맞게, 컴포넌트 변환을 실시한다.

④ 어플리케이션 개발: 후보컴포넌트를 거쳐 표준 컴포넌트 저장소에 저장된 컴포넌트를 이용하여 어플리케이션을 개발하는 과정을 수행한다.

⑤ To-Be 어플리케이션 관리: 레거시시스템에서 추출된 후보컴포넌트가 정제과정을 거쳐 재사용 가능한 컴포넌트로 전환이 완료되었다.

3.2.3 개발방법론의 주요 산출물

본 연구에서 개발한 Web환경 시스템 구축개발을 위한 LM개발방법론 RE-Method(v2.0)은 종래의 마르미-RE에 비해 각 단계 별로 상기단계의 문제점을 해결하기 위한 추가된 산출물은 [표 3], [표 4], [표 5]와 같다.

표 3. 계획단계에서 추가된 산출물과 내용

산출물	내 용
•시스템 구성도	레거시시스템에서 사용하는 하드웨어, 시스템 소프트웨어, 응용프로그램, 데이터 등의 시스템 구성요소들을 도식화한 산출물
•비전 문서	프로젝트 관련 당사자들을 정의하고, 그들을 이해 시키기 위한 레거시시스템의 문제점과 개선방안을 구체적으로 제시하며 최종 개발요소들의 지원기능 및 효과를 기술하는 산출물
•개선 아키텍처 문서	시스템의 구성요소 및 그것들의 인터페이스와 협력요소 등에 의한 시스템의 구조적 상관관계를 표현하기 위한 것으로, 레이어 레벨의 업무와 비즈니스 간의 개괄적인 형태를 정의한 산출물

표 4. 역공학단계에서 추가된 산출물과 내용

산출물	내 용
•변수 관계표	레거시시스템에서 정의되거나 사용된 변수들을 식별하고, 이들의 의미 파악을 위해, 프로그램 이름 및 이에 포함된 레코드 이름과 데이터 파일, 키 등을 서술
•재구조화된 코드	레거시코드로부터 중복되거나 사용되지 않는 코드를 제거 고, 논리적으로 복잡한 제어 구조들을 FOR 문 PERFORM 문 등을 통해 체계화한 코드 산출물
•서브루틴 호출 그래프	레거시프로그램을 구성하는 서브루틴 상호간의 정적인 호출흐름 관계를 모형화하기 위해, 계층적 방향그래프 형식으로 표현
•시스템 자원 그래프	시스템수준에서 레거시프로그램에 사용된 물리적 자원들을 표현한 그래프로, COBOL의 JCL, MAPSET, COPYBOOK, FILE, DB 등의 요소가 표현
•화면 흐름 그래프/표	레거시시스템의 화면흐름을 파악하여, 단위 기능별로 워크플로우를 식으로 하고 이를 표 또는 그래프로 나타낸 산출물
•구조적 아키텍처	레거시시스템에 대한 프로그램 분석 및 설계 정보 이해 활동을 통해 추출된 정보들을 기반
•행위적 아키텍처	레거시시스템의 전체적인 행위를 파악하기 위해 구조적 아키텍처를 구성하는 서브시스템들을 바탕으로 호출관계를 동적으로 표현한 산출물
•레거시 기술 아키텍처	레거시시스템을 구성하는 하드웨어 장비와 이러한 장비에 배치된 서브시스템들이 어떤 기술들이 적용되었는지를 기술

표 5. 컴포넌트단계에서의 추가된 산출물과 내용

산출물	내 용
•컴포넌트 목록표	레거시시스템으로부터 컴포넌트 마이닝 활동을 통해 추출된 컴포넌트들에 대한 정보를 기록
•컴포넌트 구성 요소 기술서	레거시시스템으로부터 획득한 컴포넌트 각각에 대하여 컴포넌트를 구성하는 시스템 요소들에 대한 정보와 그들 간의 연관관계 및 컴포넌트 운영환경과 필수 요구 조건과 제약 사항 등 에 대해 기술
•컴포넌트 상호 작용표	레거시시스템으로 부터 추출된 컴포넌트들간의 연관관계나 상호작용에 대한 정보를 제공
•아키텍처 품질 속성 목록표	목표시스템에서 실현해야 할 품질 속성들을 식별하고 연관된 다른 속성들과의 부작용 관계정보
•소프트웨어 아키텍처	시스템의 비기능적 요구사항 및 품질속성들을 고려하여 목표시스템에 대한 주요 결정 사항들을 파악하고, 이를 목표시스템 상으로 반영할 수 있도록 참조하는 아키텍처 스타일에 추출된 후보 컴포넌트를 배치
•시스템 아키텍처	목표시스템의 물리적인 환경요소들을 명시하는 기술 아키텍처 및 추출된 컴포넌트의 연관관계에 따른 컴포넌트간의 계층적구조를 나타내는 컴포넌트 아키텍처를 정의
•유스케이스 명세서	계획단계와 역공학단계를 통해 추출한 유스케이스 각각에 대하여 유스케이스 내의 작업흐름을 기술

3.2.4 개발방법론과 지원도구 인터페이스

개발방법론의 적용빈도를 높이고 개발기간 단축과 표준화 및 효율성을 높이는 방법으로 개발 프로세스 부분에서 자동화가 가능한 부분을 집중적으로 개발하는 작업이 반드시 선행되어야 한다. 과거 국내에서 많은 개발방법론이 발표되었지만 실제 사용자들이 외면하는 문제점을 살펴보면 첫째, 개발기간이 오히려 많이 걸리며 둘째, 프로세스를 위한 작업이 많고 셋째, 방법론과 연결된 적합한 지원도구를 찾기 힘들다는 점이다.

본 연구에서는 이러한 문제점을 보완하고 실제 적용을 통하여 학문적 이론과 연결시키는 방법을 전개하였다. 여기서 개발방법론을 효과적으로 지원하기 위해 지원도구의 지원이 가능한 역공학단계와 컴포넌트화 단계 부분에 대하여 연결할 수 있는 구체적 방안에 대하여 기술하였다.

(1) 프로그램 분석(파싱)

[그림 6]은 계획단계에서 작성된 개발계획서에 따라 처음단계로 접어들면서 레거시시스템의 프로그램(비즈니스 로직, 데이터접근, 사용자 인터페이스부분)을 정밀하게 분석하는 과정에 접어들게 된다.

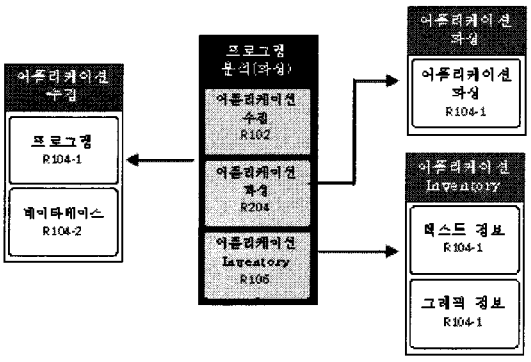


그림 6. 프로그램의 파싱도

• 어플리케이션 수집 활동: 프로그램과 데이터 부분의 자료를 수집하게 되며, 다양하게 산재해 있는 자료를 빠짐없이 수집하는 활동이 가장 중요하게 된다. 수집된 원시 프로그램의 정보의 모든 내용을 지원도구에

의하여 컴퓨터 수집저장소에 카타로그 부분과 소스 내용을 연결하는 인덱스 정보를 만들어 준다.

- 어플리케이션 분석(파싱) 활동: 수집된 어플리케이션 정보를 기반으로 지원도구에 의하여 분석작업을 거치게 되는데 구문분석과 의미분석을 통해 프로그램과 데이터베이스 그리고 JCL, Copybook, Map등과 같은 어플리케이션 구성요소간의 연결관계를 분석한다.
- 어플리케이션 인벤토리 활동: 분석된 정보를 기반으로 어플리케이션의 인벤토리를 자동으로 작성하게 된다. 이와 같은 목적은 수집한 As-Is시스템의 현황정보의 적합성과 누락부분이 있는지, 중복조사 등 수집정보의 정당성을 확인한다.

(2) 프로그램 분석

[그림 7]은 As-Is시스템을 수집하여 인벤토리작업을 거쳐 저장된 인벤토리저장소를 통하여 정밀 분석작업에 접어들게 된다.

- 의미정보 등록: 기존 어플리케이션은 수작업(혹은 지원도구)을 통해 As-Is시스템을 역공학 기법으로 역전개하기 위한 기초자료 즉, 의미정보를 등록하게 되면 실제로 사용자 분석 및 설계정보를 자동으로 출력이 가능하다. 이를 위해 프로그램 및 데이터베이스에 필요한 설명 혹은 주석부분의 정보를 자동으로 추출하여 인용한다. 또한 프로그램 내부에 주요한 문장 그룹(혹은 파라그래프)에 대한 사용자가 이를 입력함으로 자동 분석시에 기술정보에서 더욱 구체적인 사용자 분석정보를 출력하게 된다.
- 프로그램 분석정보: 업무구조도, 업무흐름도 등의 다양한 시각화정보를 제공함으로써 사용자는 복잡한 소스코드나 업데이트되지 않은 문서의 활용을 억제할 수 있다. 즉, 실제 운영되는 프로그램 정보를 기반으로 다양한 시각화 정보를 제공함으로써 시스템의 정확도를 한층 높여주는 계기가 된다.
- 시스템구조 분석정보: 업무모듈도, 업무관련도, 업무-프로그램-데이터 간의 메트릭스 정보, 시스템 사용빈도, 그리고 영향분석 정보를 제공하게 된다. 업무 간의 연결관계와 데이터베이스와 업무 간 혹은 프로그램간의 자료처리 형태 등의 정보를 제공하며, 유지보수하는데 가장 중요한 기능인 영향분석정보는 매우 유익한 정보로 활용된다.
- 업무기능 분석정보: 역공학을 통하여 As-Is정보를 개발자-설계자-분석가 등의 3레이어 별로 정보의 제공이 가능한 기능을 수행한다. 즉, 업무-기능간의 업무흐름도(DFD, Data Flow Diagram), 기능 간 흐름도 정보 등을 제공한다. 향후 엔터프라이즈 아키텍쳐 기능 중에 어플리케이션 아키텍처 기능의 정보를 제공한다.
- 통계정보 및 메타정보관리: As-Is시스템의 다양한 통계정보와 수집된 정보와 이를 다양한 분석절차를 거쳐서 메타데이터 저장소에 정보를 담아두며 이를 다양한 Query를 통하여 관리 혹은 자료를 자동으로 출력이 가능하도록 설계하였다.

(3) 데이터 분석

어플리케이션의 한 축인 데이터 부분의 정밀 분석작업의 단계이다. 데이터 카타로그 정보 만으로 데이터베이스

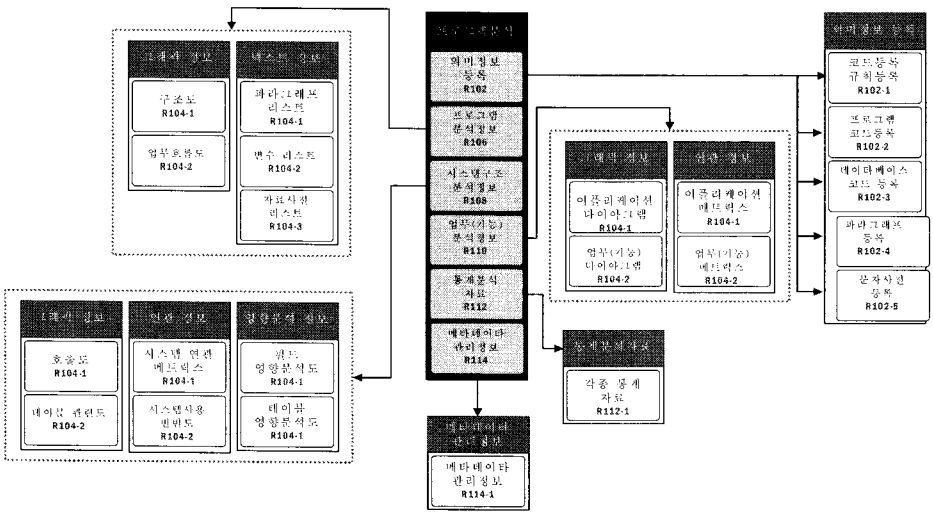


그림 7. 프로그램 분석도

정보를 추출하여 이를 Entity Relation Diagram(이하 ERD)을 구축하며 이를 자동으로 스키마를 생성하는 기능을 제공하게 된다.

- 데이터 마이닝: 데이터베이스 카타로그 즉, 데이터베이스 필드, 레이아웃 등의 정보를 자동으로 추출하여 저장한다.

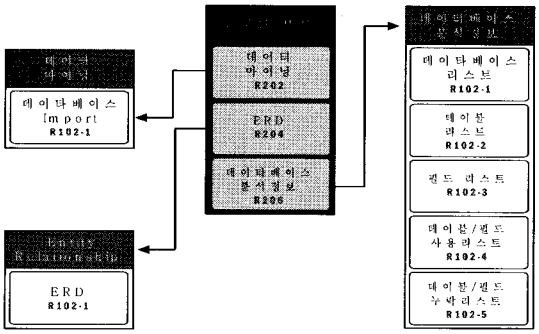


그림 8. 데이터 분석도

- 엔터티 연결도: 추출된 데이터정보를 기반으로 브라우져를 통하여 자동화된 도구에 의하여 엔터티 간의 연결관계를 나타낸다. 또한 이들의 연결관계를 쉽게 파악하며, 이를 수정하거나 연결관계를 새롭게 추가할 수 있다.
- 데이터베이스 정보분석: 추출되어 연결관계를 새롭게 한 후에 이들의 데이터 정보에 관한 여러 가지 정보를 자동으로 출력하는 정보를 제공한다.

(4) 프로그램 재구조화

프로그램 재구조화란, 비구조화된 프로그램언어들을 컴포넌트 추출이 원활하게 하기 위한 사전 정리작업이라고 정의할 수 있다. LM을 위한 프로그램언어의 구성을 3가지로 분류가 가능하다. 첫째는 메인프레임 하에서 작성된 프로그램 언어인 COBOL, PL/I, C, Assembler 와 같이 구조화된 프로그램 언어가 아닌 형태가 재구조화 대상이 된다.

- 코드제거: As-Is 어플리케이션은 체계화, 표준화에 염두에 두고 작성된 소스코드가 아니기 때문에 불필요하나 중복작성, 프로그래머 자신만이 아는 어떠한 주석문 등으로 인해 복잡하게 되어있어 불필요한 부분을 제거하는 작업이다.
- 코드정제: 정리된 소스코드를 새로운 코딩 규칙에 맞게 재 작성하거나 표준 포맷에 맞게 기존의 소스코드를 표준화된 규칙에 맞게 재정리하는 작업을 하게 된다. 여기서 비구조화에 역행되는 프로그램 언어의 문장 즉 Go to, Perform, Link, XCTL과 같은 프로그램 구조를 제어하는 문장 등을 비주얼 차트로 소스코드를 조명한 후에 이를 조정하거나 대체하는 작업단계이다.
- 연관분석: 이미 프로그램 언어의 영향도 분석작업을 통해 연관정보, 영향정보 등은 메타데이터에 저장되어 있으나, 가장 현실적으로 사용하는 프로그램 내에서의 연관분석(파라그래프 혹은 모듈 간)정보를 파악하는 작업을 거치게 되는데 그 목적은 프로그램 소스코드를 재조정하는데 가장 중요한 작업이 필요하기 때문이다. 반드시 정리된 이전과 이후의 이력정보를 자

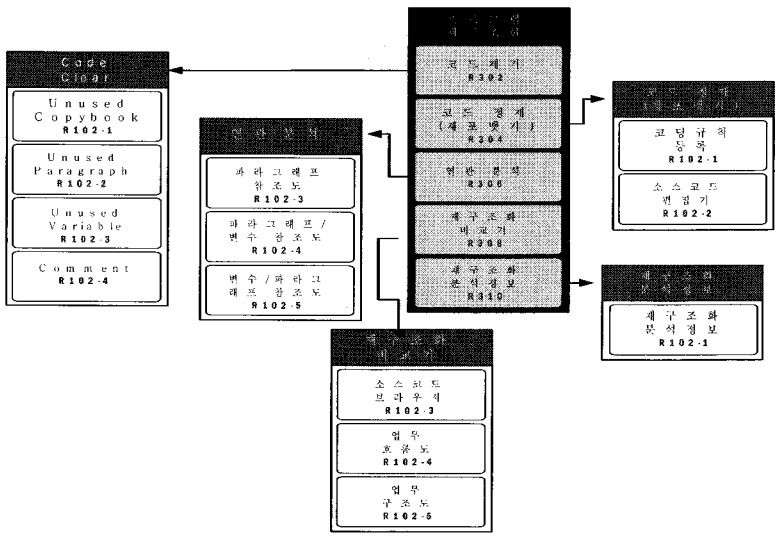


그림 9. 프로그램의 재구조화

동으로 작성해야 한다.

- 재구조화 비교기: 전 단계에서 이력정보 작성을 하였기 때문에 전후의 소스코드를 비교하는 기능을 제공한다.
- 재구조화 분석정보: 재구조화된 정보에 관한 여러 가지 정보를 자동으로 출력하는 정보를 제공한다,

(5) 후보 컴포넌트 식별

후보 컴포넌트(Candidate Component)는 작은 알고리즘으로 구성된 컴포넌트에서 시작하여 어플리케이션 패키지 그리고 비즈니스 전체를 컴포넌트로 보는 시각이 있으나 본 연구에서는 프로그램 단위를 컴포넌트 대상으로 삼고 연구하였다, 이를 좀 더 세밀하게 정리를 하여 사용자 인터페이스에 해당하는 경계컴포넌트, 데이터를 제어하는 엔티티컴포넌트 그리고 비즈니스 로직에 해당하는 제어컴포넌트로 구성하였다.

[그림 10]은 하나의 프로그램(혹은 어플리케이션) 언어를 분류한 그림이다. 이러한 관점에서 레거시시스템에서 재사용 대상이 되는 후보 컴포넌트 추출 프로세스가 진행이 된다.

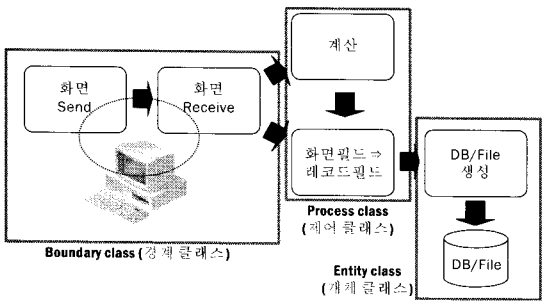


그림 10. 후보 컴포넌트 추출 범위

- 후보 컴포넌트 추출: 후보 컴포넌트 분류작업은 [그림 11]와 같이 3가지의 컴포넌트(혹은 Class)를 추출하는 작업을 수행한다.
- 후보 컴포넌트 정제: 추출된 후보 컴포넌트 역시 목표 프로그램언어의 구조가 아닌 레거시시스템의 형태의 소스코드를 정제하는 작업을 수행한다.
- 후보 컴포넌트 등록: 이렇게 분리된 후보 컴포넌트를 재사용하기 위해서는 이미 저장 관리되고 있는 재사용 가능한 컴포넌트 저장소의 컴포넌트와 비교 매핑하여 교체, 신규등록, 사용불가 등을 위한 작업을 하게 된다.
- 후보 컴포넌트 분석정보: 후보 컴포넌트에 대한 다양한 분석 정보를 제공한다.

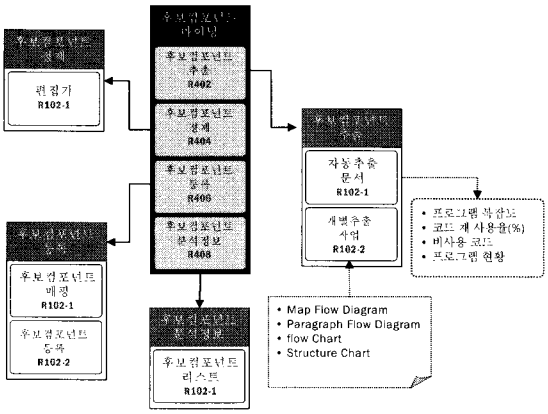


그림 11. 후보 컴포넌트 추출

4. LM 적용결과

4.1 LM 적용의 효과

이 부분은 LM 방법론을 적용했을 때 획득할 수 있는 효과를 정성적 효과와 정량적 효과로 나누어 분석하였다. 2001년부터 6년여간 외환은행, KT 및 일본 JRI, APSA, 미국, 산마태오 정부 등, 92개 업체를 대상으로 RE-Method(v2.0)와 적용해 본 결과, 다음과 같은 정성적/정량적 기대효과를 얻을 수 있었다.

(1) 정성적 효과

첫째, 관리적 측면의 기대효과이다.

경제성 : LM 개발방법론과 절차에 따라 지원도구를 활용함으로 정확한 모델링과 의사소통으로 생산성을 향상 시키고 프로젝트에 소요되는 시간과 비용이 절감이 된다.

품질향상 : 체계적인 방법론과 지원도구를 활용하는 것은 기존의 수작업에 의하여 비 표준화된 상황과는 달리 업체내의 표준화와 절차의 합리화를 통하여 품질 향상의 기대를 얻을 수 있다.

생산성 : 수작업으로 분석/설계 할 경우 보다 100% 이상의 생산성 효과가 발생하고(LM지원도구의 분석기능을 활용하면), 개발 전 작업과정 평균 2.3배의 생산성효과로 분석되었다.

효율적인 역할 담당 : 개발방법론에는 프로젝트 참여자의 다양한 역할과 책임이 부여되어 업무의 효율적인 전개와 역할을 수행하게 된다.

소요인력의 감소 : 단순 반복적인 기술 부분은 자동화된 지원도구가 처리함으로 적재적소의 인력을 배치하고 경력관리가 용이하다.

문서관리 : Legacy 시스템의 사장된 기존의 문서 출력 방식에서 자동화된 지원도구에 의하여 이력관리, 형상관리, 실시간 출력 등의 기능을 이용하여 정확하고 빠른 문서를 출력하게 된다.

유지 보수 비용절감 : LM 개발방법론과 지원도구 활용으로 체계화된 새로운 플랫폼의 시스템을 쉽게 접근이 가능하고 자동화 처리로 인해 유지 보수가 정확하고 빠르고 유연하게 대처함으로 비용절감 효과가 분석되었다. 수작업에 비해 20-30%의 개발생산성 효과는 표준화된 CBD기반의 지원도구를 활용함으로 얻어지게 분석되었다.

둘째, 기술효과적 측면의 기대효과이다.

경제성 : LM 개발방법론과 절차에 따라 지원도구를 활용함으로 정확한 모델링과 의사소통으로 생산성 향상을 시켜 프로젝트에 소요되는 시간과 비용을 절감하는 효과가 분석되었다.

기술습득 : 표준화된 절차와 지원도구를 통해 컴포넌트 기반의 신기술에 쉽게 적응이 가능하고 기술을 쉽게 배울 수 있는 것이 분석되었다.

지식관리 : 시스템 개발의 일관성을 확보하고 지식의 체계화와 지식관리를 통하여 용이성, 우지보수성, 기능향상, 품질향상 등의 효과가 분석되었다. 이는 메타제이타, 컴포넌트 저장소, 어플리케이션 정보 저장소 등의 지식이 등록이 되어 기존의 특정한 공급자의 제품에서 독립적이라는 점이 효과로 분석이 되었다.

개발기간 준수 : 프로젝트의 효율적 관리로 개발기간 단축은 물론 완료시기를 정확하게 준수하는 효과로 분석이 되었다.

자원의 재활용 : 기존 전산인력의 활용으로 인력교체의 불안감을 해소하고 기존 업무 프로세스의 자동 승계로 업무 프로세스 변경에 대한 불안감이 해소가 되며, 유지 보수 비용절감으로 수익성이 증대되는 것으로 분석이 되었다.

(2) 정량적 효과

정량적 효과는 우선 Legacy 시스템을 다양하게 적용한 표준 템플릿의 효과와 세가지 업종에 따른 유형별로 효과 부분을 나누어 연구하였다. [표 6]은 작업 구분별 생산성 비교 결과를 나타내고 있다.

[표 7]은 정부기관, 제1, 2금융기관 등 대표적인 업체를 선정하여 이를 LM을 적용한 결과이다. 이 적용사례에서 시사하는 바는 여러 가지로서 우선 LM지원도구를 적용한 결과와 그렇지 않은 결과가 업체마다 다르다는 점이다. 여기에는 레거시시스템의 표준화문제 등이 변수로 작용이 되었거나, 언어가 갖는 다양성과 특징에 따라 차이가 있음을 볼 수 있다. 즉, 정량적 효과는 LM개발 생산성 2.3배, 평균 생명주기 연장효과 110% 그리고 레거시 시스템 분석 공수의 10% 감소, 개발기간의 20% 감소, 유지보수 비용을 30%이상 절감효과를 볼 수 있으며, 기술습득 기간도 기존의 수작업 대비하여 10% 이상 단축되는 것으로 나타났다.

표 6. 작업 구분별 생산성의 비교

작업구분	처리내용	신규개발시	Re-Method적용	효과
처리	표준 배치 작업	1본/4일	1본/2일	50%절감
등록	유지보수, 관리화면 참조	1본/4일	1본/2일	50%절감
조회	단순한 조회(Without logic)	1본/2일	3본/1일	83% 절감
출력Form		2본/3일	1본/1일	33%절감
출력일반		1본/1일	2본/1일	50%절감

표 7. LM의 적용사례 결과

사례	볼륨(본)	구분	전환율(%)	기간(개월)	소요공수 (M/M)	비용(백분율)
공공기관	627	비적용	-	7	30	100
		툴 적용	77.5	2	14	46.6
		효과	-	5(-71.5%)	16(-53.4%)	-53.4%
제1금융기관	3150	비적용	-	18	260	100
		툴 적용	92.3	8	180	69.2
		효과	-	10(-60%)	207(-30.8%)	-30.8%
제2금융기관	1106	비적용	-	10	179	100
		툴 적용	85.6	7	95	53.1
		효과	-	3(-30%)	84(-46.9%)	-46.9%

표 8. 마르미-RE와 RE-Method의 비교

구분	마르미-RE			RE-Method		
개발시기	2002-2003년			2004-현재		
적용범위	- 컴퍼넌트화 개발방법론			- LM(Legacy to Modern) 개발방법론		
대상언어	- COBOL			- Java 환경 / EJB		
자동화 도구 지원 여부	- 지원 불가능			- 단계별 지원 가능		
분석기능	- 컴퍼넌트 추출 정도			- 다양한 자동화 문서 제공 (영향분석/변경관리/중요도관리/시스템 자원간의 인터페이스)		
방법론 수준						
단계구성	단계	활동	작업	단계	활동	작업
	계획단계	2	4	계획단계	4	11
	아키텍처 단계	3	6			
	점진적 개발단계	2	7	역공학단계	5	19
	인도단계	2	4	컴퍼넌트화 단계	5	13

4.2 LM 방법론의 장점 분석

2003년에 발표한 '마르미-RE'가 현재까지 보완된 RE-Method(v2.0)의 비교 및 구체적으로 추가한 차별성과 특징적 기능은 [표 8]과 같이 요약할 수 있다.

5. 결 론

본 연구는 LM구축을 위한 개발방법론을 제안하고 이를 적용한 결과에 관한 것으로 Y2K 지원도구의 개발과 이를 통한 사례와 레거시시스템 분석경험, 소프트웨어 재사용에 관한 연구들을 분석하고 Web 환경시스템 구축 개발을 위한 레거시시스템을 재사용하기 위한 개발방법론을 설계하였다.

레거시 마이그레이션 개발방법론은 레거시시스템을 재사용할 수 있도록 하여 기존 업무프로세스의 승계로 업무 프로세스 변경의 불안감을 해소하며, 유지보수 비용 절감으로 수익성의 증대와 생산성 향상을 얻을 수 있으며, 수작업 대비 20-30% 유지보수 향상효과를 얻을 수 있다.

이와 같은 연구결과를 통하여 LM에 관한 후속 연구를 제언하면 다음과 같다.

첫째, 개발방법론의 지속적 발전이 필요하다. 즉, 레거시시스템에서 재사용 컴포넌트 정보를 추출하는 과정은 비정형자료 추출에 해당하여 수많은 변수가 작용하게 된다. 이러한 점으로 이를 처리하는 프로세스 또한 변화를 하면서 대응하지 않으면 안 될 것이다. 마르미-RE에서 RE-Method를 거치면서 이를 지속적으로 계승 발전하여야 품질 좋은 컴포넌트를 추출할 수 있으며 절차의 최적화를 통해 개발기간 단축의 효과를 얻을 수 있을 것이라고 판단한다. 둘째, 개발방법론과 동일하게 지속적으로 개발방법론에 맞추어 자동화 비율을 높이는 지원도구를 개발하여야 할 것이다. 셋째, 사장되어가는 레거시시스템을 지속적으로 유지보수의 품질을 향상시켜 레거시시스템의 지적재산을 보호하여야 한다.

참고문헌

[1] Chuck Martin, "e-비즈니스.com", 21세기북스, 1999.

[2] 김진우, HCL Lab, "인터넷 비즈니스", 영진출판사, 1999.

[3] 박병형, "밀레니엄 버그", 영진출판사, 1998.

[4] 박병형, "블록놀이와 CBD Component Based Development", 태영출판사, pp14-17, 2002.

[5] 박병형 저, "한권으로 끝내는 e-ERP", 태영출판사, pp25-27, 2001.

[6] E. J. Chikofsky Reverse Engineering and design recovery, "A taxonomy". IEEE Software, 7(1):13-17, 1990.

[7] M. Fowler, Refactoring, "Improve the design of existing code". Addison-Wesley, Reading MA, 1997.

[8] National Information Standards Organization, "Understanding Metadata", NISO Press(ISBN:1-880124-62-9), 2004.

[9] H. M. Sneed, "Object Oriented Software migration". Addison -Wesley, German, 1998.

[10] H. A. Muller, M. A. Orgun S. R. Tilley and J. S. Uhl. "A reverse engineering approach to subsystem structure identification". Journal of Software Maintenance, 1993.

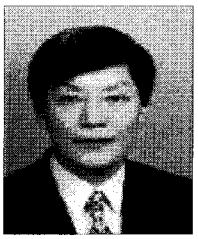
[11] Frost & Sullivan, "Frost & Sullivan White Report", Palo Alto, USA, 1999.
[12] 박병형, "레거시시스템의 컴포넌트화 방법론 "마르미-RE", Magic and Robust methodology Integrated-Reengineering(v1.0), 주관기관:한국전자통신연구원, 참여기관:(주)케미스, 2003.
[13] 양해술, 박병형, "CBD 환경을 위한 4GL 어플리케이션의 웹어플리케이션으로의 변화", 한국정보처리학회지. 1226-9182, 제10권 제3호, pp89-96, 2003.
[14] 박병형, 이승호, 김영희, "MSF/CD 방법론 기반의 분석/설계 도구설계에 관한 연구", 한국정보처리학회지, 1226-9182, 제10권 제3호, pp63-73, 2003.
[15] 박병형, 이은성, 이승호, "MSF/CD를 기반으로 한 ERP솔루션 개발사례", 한국정보처리학회지, 1226-9182, 제10권 제3호, pp137-146, 2003.
[16] 최일우, 류성열, "레거시시스템 진화를 위한 효율적 재공학 프로세스", 한국정보처리학회 논문지D, 제10-D권 제5호, pp845-858, 2003.
[17] Dolly M. Neumam, "Evolution process for Legacy system Transformation", ACM, 1998.
[18] Rober C. Seacord, John Robert, "A survey of legacy system Modernization Approaches", Technical. Note CMU/SEI-2000-TN-003, April, 2000.

이 준 웅(Jun_Woong Lee) [정회원]

- 1968년 : 성균관대학교 행정학과 졸업(행정학사)
- 1975년 : 서울대학교 행정대학원 졸업(행정학석사)
- 2005년~현재 : 호서대학교 벤처전문대학원 정보경영학과 박사과정
- 1975년~77년 : 명지대학 행정학과 강사
- 1978년~93년 : 국제그룹 및 한일그룹 관리 본부장
- 1993년~현재 : 국제경영연구원 대표/경영지도사
- 2006년~현재 : (사)한국경영.기술컨설턴트협회 강남지회장

<관심분야>
품질 및 정보경영, 기술컨설팅, 개발 및 관리방법론, 프로젝트 및 품질관리

양 해 술(Hae-Sool Yang) [정회원]



- 1975년 : 홍익대학교 전기공학과 졸업(학사)
- 1878년 : 성균관대학교 정보처리학과 졸업(석사)
- 1991년 : 日本 오사카대학 정보공학과 소프트웨어공학 전공(공학박사)
- 1975년~79년 : 육군중앙경리단 전자계산실 시스템분석장교
- 1980년~95년 : 강원대학교 전자계산학과 교수
- 1986년~87년 : 日本 오사카대학교 객원연구원
- 1994년~95년 : 한국정보처리학회 논문편집위원장
- 1995년~2002년 : 한국S/W품질연구소 소장
- 2001년~현재 : 한국정보처리학회 부회장
- 1999년~현재 : 호서대학교 벤처전문대학원 교수

<관심분야>
소프트웨어공학(특히, S/W 품질보증과 품질평가, 품질감리 및 컨설팅, OOA/OOD/OOP, SI), S/W 프로젝트관리, 컴포넌트 기반 개발방법론과 품질평가