제18회 한국정보처리학회 추계학술발표대회 논문집 제9권 제2호 (2002.11)

레가시 시스템의 컴포넌트화 방법론 개발을 위한 프로그램 분석 활동

차정은*, 김철홍*, 양영종*
*한국전자통신연구원, S/W · 컨텐츠 연구부
e-mail:{mary2743, kch, yangyj}@etri.re.kr

Program Analysis Activities for Development of Componentization Methodology for of Legacy System

Jung-Eun Cha Cheol-Hong Kim Young-Jong Yang
ETRI Computer Software Tech. Lab, S/W · Contents Tech. Department

요 약

기업의 비즈니스 프로세스가 복잡, 다양해짐에 따라, 현재 운영 시스템에 대한 급격한 기술적 변화를 수용하고 이를 조직적 측면의 기업 프로세스로 적용하기 위해 레가시 시스템의 현대화가 요구된다. 따라서, 현재의 기업들은 다양한 사용자들이 각자 그들의 관점에서 필요한 비즈니스 요구들을 웹 상에서 처리시킬 수 있도록 J2EE, .NET 등으로 대표되는 컴포넌트 및 웹 서비스 기술을 적용한 새로운 e-business 환경을 수용해야만 한다. 하지만, 기업 조직의 중요한 지식과 프로세스들을 처리하는 시스템들은 대부분 과거(Legacy)의 기술에 의해 개발되어졌으며, 이러한 시스템들을 새로운 비즈니스 환경에 적용하기에는 웹 환경을 위한 분산 아키텍처의 결여와 개방성과 표준화 미흡으로 시스템의 유지보수에 많은 어려움을 가진다. 또한, 방법론 차원에서 재공학의 절차와 기법을 체계적으로 정의하고 지원하기 위한 노력이 매우 부족한 실정이다.

본 논문에서는 레거시 시스템을 새로운 시스템 환경으로의 변환 및 통합을 위한 재공학 방법론 개발을 목적으로 프로그램 분석 활동을 설명한다. 본 논문에서 개발하고자 하는 방법론은 다양한 추상화 수준에서 역공학 정보를 복구하고, 컴포넌트화 단계를 통해 새로운 시스템으로 진화할 수 있는 절차 및 기법들을 제공한다. 레거시 시스템 컴포넌트화 방법론은 변환계획 단계, 역공학 단계, 컴포넌트화 단계, 인도 단계의 4 단계로 구성되어 있으며, 본 논문에서는 전체 단계 중 가장 기초가 되고 중요한 단계인 역공학 단계에 초점을 두고 프로그램 분석을 위한 절차 및 과정의 주요 지침들을 제시한다.

1. 서론

기업의 비즈니스 프로세스가 복잡, 다양해짐에 따라, 현재 운영 시스템에 대한 급격한 기술적 변화를 수용하고 이를 조직적 측면의 기업 프로세스로 적용하기 위한 패러다임의 전환이 요구된다. 특히 인터넷 환경의 보편화 및 웹 서비스, 컴포넌트 기반 개발 등의 새로운 기술들의 도입, 기업 응용들 간의 통합 필요성 등은 다양한 사용자들이 각자의 관점에서 필요한 비즈니스 요구들을 웹 상에서 처리시킬 수 있는 e-business 환경으로의 변환을 더욱 요구한다. 종래의 레가시 시스템들은 웹 환경을 위한 분산 아키텍처가 결여되어 있으며, 개방성과 표준화 미흡으로 시스템 자체의 융통성이 매우 부족하다. 또한 사용자 지향의 친밀한 인터페이스를 통해 상호 대화적 방식을 통한 동적인 의사 결정이 불가능하다[1]. 그러므로, 이러한 문제의 해결을 위해서는 레가시 시스템의 이해를 극대화하여, 시스템이 가지는 제한성들을 해결해야 한다. 또한 장기간에 걸친 레가시 시스템의 진화 모델 확립함으로써, 유지보수성을 향상시키고, 새로운 요구에 대해 융통성 있게 대응하기 위해 레가시 시스템의 현대화 기술이 필요하다. 지금까지, 방법론 차원에서 재공학의 절차와 기법을 체계적으로 정의하고자 하는 노력은 미흡하였다. 따라서 많은 기업들이 재공학 프로젝트를 추진함에 있어서 시행 착오를 겪고 있다. 본 연구에서는 레거시 시스템을 컴포넌트 기반 시스템으로의 체계적 변환 및 통합을 위한 프로세스와 기법을 제공하고자 한다. 현재 1단계 연구가 진행 중이며, 본 논문에서는 전체 재공학 방법론의 프로세스를 서술하고, 그 중 가장 기초적인 단계인

<표 1> 레가시 변환 기술의 비교

방법		설 명
객체지향 프레임워크 변환	특징	레가시 코드를 분석하고 업무 지식을 추출하며, 자동적인 변환을 통해 표준화된 객체나 컴포넌트로 패키징
	장점	레가시의 분석/모델링에 효율적,탄력적인 솔루션 제공
	단점	레가시 영역의 제한성 및 자동화에 따른 성능 저하
랩핑 (Wapping)	특징	컴포넌트로의 변환 위해 OPEN API 인터페이스 제공
	장점	분산 아키텍처 지원, 레가시 코드로부터 컴포넌트 생성 용이
	단점	제한된 융통성에 의한 유지보수 문제 발생
스크린 스크랩핑 (Screen Scraping)	특징	레가시의 UI를 추출하여 웹 상의 스크린으로 변환
	장점	UI 수준에서 웹으로 변환하기 위한 가장 빠르고 간단함
	단점	실질적 변환이 아니며, 기능성과 융통성이 제한됨
미들웨어	특징	시스템간 매핑 통해 레가시 시스템을 웹과 응용 서버에 연결
	장점	트랜잭션 수준 통합으로 많은 벤더에 의해 워크플로우 지원
	단점	다른 코드와 연관해 사용하며, 프로세스 지식 확보가 불가능
EAI (Enterprise Application Integration)	특징	특정 컨넥트를 통한 레가시와 웹 응용 사이의 시스템 통합
	장점	레가시 기능들을 바탕으로 벤더에 의한 제품 지원이 다양
	단점	레가시 시스템과의 중복된 투자와 높은 비용 요구

역공학 단계에서 수행하는 레가시 프로그램의 분석을 위한 절차와 기법, 주요 지침 사항들을 제시하는 프로그램 분석 활동을 상세히 서술한다.

2. 관련연구

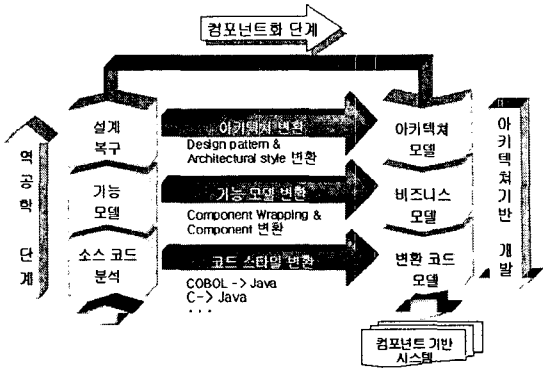
2.1 레가시 시스템의 재공학

재공학은 레가시 시스템에 대한 분석과 재구조화를 통해 다양한 관점에서의 형상을 식별하고 유지보수 정보를 획득하며, 나아가 새롭게 기대되는 기능적, 환경적 요구에 맞도록 재구축하는 광의의 공학적 접근이다. 따라서 재공학 목적이 레가시 시스템의 품질 향상 혹은 재사용성 강화인지, 아니면 관리 프로세스의 개선인지에 따라 다양한 관점에서 설명될 수 있다.

<표 1>은 레가시 시스템의 변환을 위한 접근 기술들의 특징과 장·단점을 설명한다[2]. 재공학을 위한 핵심적인 기술은 보다 호환성 있는 개방적인 시스템으로의 전환으로 크게 인터페이스를 통한 랩핑과 레가시 시스템의 실제적인 변경을 통해 새로운 시스템으로 재개발하는 두 가지 접근으로 구분된다[3].

2.2 재공학 방법론

SEI CMU에서 개발한 것으로 아키텍처와 코드 기반 재공학 도구들을 통합하기 위한 프레임워크을 제공하는 모델로 CORUM(Common Object-based Reengineering Unified ModelII)이 있다[4]. 이 모델은 재공학과 순공학 프로세스의 통합 모형을 제시하고, 상위 수준 산물과 하위 수준 산물이 서로 반복적으로 교류, 통합하면서 추상화 모델을 만들어 간다.



(그림 1) 레거시 시스템 컴포넌트화 방법론의 개념 모델

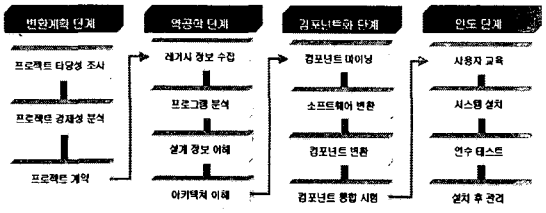
전체 4개 수준의 3개의 프로세스로, 코드와 아키텍처 모델을 복구하고 복구된 모델의 적합성 평가 프로세스와, 복구된 아키텍처를 새로운 아키텍처로 변환하고 평가하는 프로세스, 그리고 아키텍처 기반 개발 프로세스로 구성된다.

또한, MORALE(Mission ORiented Architecture Legacy Evolution)은 Georgia Institute of Technology에서 Mosaic 웹 브라우저에 새로운 요구사항을 반영하여 향상된 시스템으로 전개하는 메소드이다[5]. MORALE은 기술적 요소보다 조직의 미션(Mission)에 중점을 둔 프로세스를 제공하며, 시스템의 구조적 변경에 대한 영향을 예측함으로써 시스템 진화 초기에 변경에 대한 위험 요소를 알아낸다. 따라서 기존 시스템을 효과적으로 분석하고 새로운 시스템에서 사용할 수 있는 부품을 추출할 수 있도록 한다.

3. 레거시 시스템 컴포넌트화 방법론

본 방법론은 COBOL, RPG, PL/1, C 등으로 작성된 레거시 시스템을 J2EE, .NET 등의 컴포넌트 기반 및 웹 서비스 시스템으로 변환하거나 래핑하기 위한 절차와 기법을 제공을 목적으로 한다.

(그림 1)은 본 방법론의 개념 모델을 보여준다. 역공학 단계는 최하위 수준의 소스 모델로부터 기능 모델, 아키텍처 모델로 추상화된 정보를 복구한다. 소스 모델은 프로그램 소스를 분석하여 텍스트 및 AST 정보를 생성하여 레거시 시스템의 코드 패턴을 식별한 것이다. 프로세스 모델은 보다 추상화된 설계 정보인 시스템과 데이터의 구조도 및 단일 로직을 수행하는 워크플로우 프로세스를 나타내며, 아키텍처 모델은 기능 모델을 그룹핑하고 필터링하는 과정을 거쳐 더 추상화된 정보인 컴포넌트(서브시스템)와 그들간의 인터페이스 관계를 표현한다.



(그림 2) 재공학 방법론의 전체 단계 및 활동

컴포넌트화 단계는 역공학의 각 수준에서 조직내의 역량에 따른 변환 프로세스를 지원한다. 코드 스타일 변환은 단순히 프로그램 언어간의 변환을 지원하며, 기능 모델 변환은 래핑과 DB 스키마 변환이 전형적인 사례이며, 아키텍처 변환은 복구된 레거시 시스템 아키텍처를 새로운 아키텍처로 변환한다.

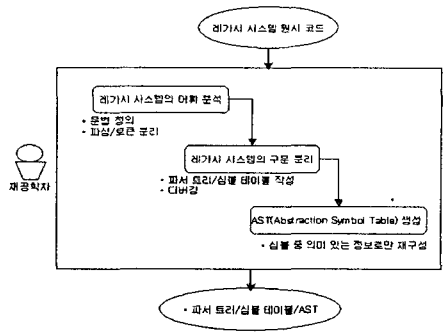
현재는 (그림 2)와 같이 구성된 전체 4 단계에 대한 1차적 프로토타입이 완성된 상태이다. 본 논문에서는 2번째 단계인 역공학 단계에서 레가시 시스템의 코드로부터 아키텍쳐 정보를 추출하기 위한 프로그램 분석 활동의 작업 절차와 기법, 고려해야할 지침에 대해 서술한다.

4. 프로그램 분석 활동

프로그램 분석 활동은 (그림 2)에서와 보는 것과 같이 원시 코드 중심의 분석 작업을 수행함으로써, 레거시 시스템에 대한 이해를 높이는 역공학 단계의 2번째 활동이다. 이 활동은 원시 코드의 분석과 재구조화를 수행하는 전형적인 역공학 프로세스이다. 따라서, 레거시 시스템의 구문과 의미 정보들을 분석하고, 코드 패턴을 식별하며, 비구조적인 코드들을 추출한다. <표 1>은 프로그램 분석 활동을 구성하는 작업들과 절차, 그리고 주요 산출물들을 나타낸 것이다. 레거시 시스템이 가진 특성들을 그대로 파악하는 것이 목적으로, 도구에 의한 자동 구문 분석 및 의미 분석을 통해 재공학 과정의 생산성을 향상시킬 수 있다.

4.1 구문 정보 분석 작업

레가시 시스템의 원시 코드를 대상으로, 시스템을 구성하는 심볼들을 분석하여 토큰으로 분리하고 구문 규칙의 확인을 통해 심볼 테이블과 파서 트리를 작성하고, 디버깅한다. 또한 심볼 정보 중, 의미 있는 정보로만을 재구성한 AST(Abstract Syntax Tree)을 작성함으로써 역공학 과정 중에 필요한 의미 정보 생성을 위한 라이브러리를 정의한다[6]. (그림 3)은 이 작업의 절차도이다.

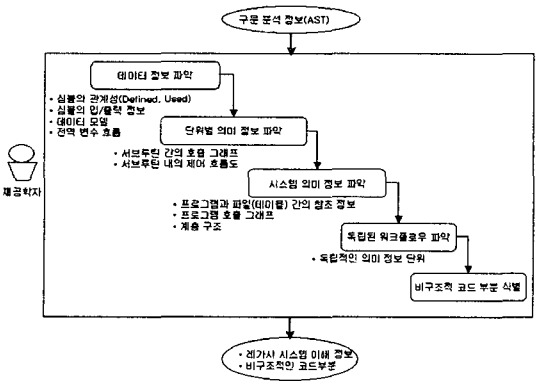


(그림 3) 구문 정보 분석 작업의 절차 구성도

4.2 의미 정보 분석 작업

이 작업에서는 시스템 전체의 구성 및 제어 흐름이 분석하고, 단위 서브 시스템 간의 호출 관계 및 데이터와 제어 흐름 등의 정보를 파악한다. 또한 독립적인 의미 단위인 워크플로우를 식별하며, 변환작업의 용이성을 위해 코드 재구조화를 수행한다[7].

(그림 4)는 이 작업의 절차도이다. 데이터 정보로는 심볼의 Define, Used 관계와 입출력 정보, DB와 연관된 데이터 모델을 식별하며, 단위 레벨 시스템 정보 파악을 위해 서브 루틴 호출 그래프, 제어 흐름 그래프 등을 이용한다. 또한 시스템 전체에 걸쳐 있는 의미 정보인 프로그램 간의 제어 흐름 및 참조 정보, 호출 관계 정보들은 프로그램(파라그래프)과 파일(테이블) 간의 참조 테이블과 프로그램 호출 그래프를 작성하여 분석한다. 또한 의미적으로 연관된 독립 단위인 워크플로우를 식별함으로써 컴퍼넌트의 후보를 제시하기 위해서는 핵심 Seed 전역 변수의 사용 범위를 찾고 그 내부에서 사용되는 데이터베이스와 분리된 인터페이스를 찾아 그룹핑한다.



(그림 4) 의미 분석 작업의 절차 구성도

<표 1> 프로그램 분석 활동의 작업 및 절차들

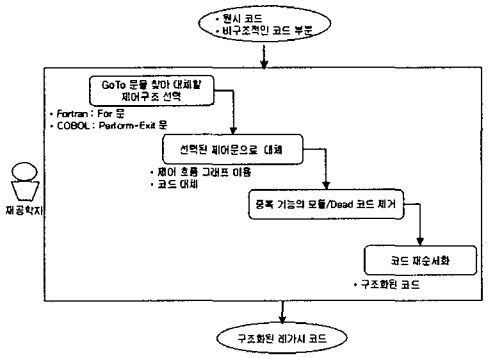
작업 번호	작업	작업 개요	절차	주요 산출물
R0202	구문 정보 분석	레거시 프로그램을 파싱하고, 분석 정보 테이블을 생성하며, Abstract Syntax Tree (AST)를 구성함으로써, 레거시 시스템의 이해 정보 추출을 위한 기초를 마련한다	① 레거시 시스템의 어휘 분석 ② 레거시 시스템의 구문 분석 ③ Abstract Syntax Tree (AST) 생성	구문 분석 결과물 · 파서 트리 · 심볼 테이블 · AST
R0204	의미 정보 분석	레거시 시스템의 이해를 향상시키는 업무를 수행한다. 이 과정에서 변수 중심의 데이터 정보와 단위 및 시스템 전체의 의미 정보과 독립된 워크플로우 정보가 파악되며, 비구조적인 코드 부분이 식별된다.	① 데이터 정보 파악 ② 단위 별 의미 정보 파악 ③ 시스템 의미 정보 파악 ④ 독립된 워크플로우 파악 ⑤ 비구조적인 코드 부분 식별	의미 정보 분석 결과물 · 데이터 정보 분석서 · 단위 의미 정보 분석서 · 시스템 의미 정보 분석서
R0206	코드 재구조화	프로그램의 로직들을 구조화시키는 업무를 수행함으로써, 레거시 시스템의 이해를 향상시키고, 재공학의 생산성 향상을 도모한다.	① 비구조적인 부분을 찾아 대체할 다른 제어 구조 선택 ② 선택된 제어문으로 대체 ③ 중복된 기능의 모듈의 제거 ④ 레거시 프로그램을 재순서화	구조화된 코드
R0208	코드 패턴 분석	레거시 시스템의 도메인에서 공통적으로 적용될 수 있는 재사용 개념들을 바탕으로 레거시 시스템에서 UI 부분, 비즈니스 로직 부분, DB 부분들을 분리한다.	① 코드 패턴 정의 ② 정의한 패턴에 맞는 코드를 찾기 위해 레거시 프로그램을 탐색 ③ 코드 패턴 분석서 작성	코드 패턴 분석서

4.3 코드 재구조화 작업

이 작업은 비구조적인 로직 부분을 구조적인 코드로 대체시킴으로써 구조적인 코드를 생성하여 레거시 시스템의 이해 증가 및 변환의 용이성을 높이는 곳이다. (그림 5)는 재구조화 작업의 절차도로, 대상 언어에 따라 적용되는 규칙들은 다르게 적용된다.

4.4 코드 패턴 분석 작업

레거시 시스템의 일반적인 요소들을 구현하는 알고리즘, 데이터 구조, 라이브러리 등을 코드 패턴으로 식별함으로써, 레거시 시스템의 도메인에서 공통적으로 적용될 수 있는 공통의 재사용 개념들을 파악하는 작업이다. 레가시 코드 패턴들을 사용자 인터페이스, 비즈니스 로직, 데이터베이스 관련 부분으로 분류하기 위해 언어별 특징을 바탕으로 CRUD(Create, Read, Update Delete) 메소드를 이용한다.



(그림 5) 코드 재구조화 작업의 절차 구성도

5. 결론

본 논문에서는 레거시 시스템을 컴포넌트 기반 시스템으로의 체계적 변환 및 통합을 위한 프로세스와 기법을 제공하고자 개발 중인 레거시 시스템 컴포넌트화 방법론 중 역공학 단계의 프로그램 분석 활동의 작업 절차와 지침을 서술하였다. 역공학 단계는 레거시 시스템 이해를 위한 가장 어렵고 중요한 단계로서 활동의 효과적인 수행을 위해서는 레거시 시스템과 관련된 도메인의 참조 지식과, 자동화된 도구의 지원이 요구된다. 본 논문을 통해 시스템 변환을 위한 분석 정보의 종류 및 분석 절차, 적용 가능한 메소드 등을 제시함으로써 재공학을 위한 체계적 접근 방식을 제공하였다. 향후, 프로토타입 형태의 본 방법론을 실제적인 비즈니스 시스템 상으로의 시범 적용을 통해 수정, 보완해 나갈 계획이다.

참고문헌

[1]. Hurwitz Group, "Integrating Your Business with The Internet: Effectively Transforming Legacy System Assets into Flexible Business Servers", Relativity Technologies, White Paper, Sep 1999.

[2]. SEI Reengineering Center, CMU, "Perspectives on Legacy System Reengineering, 1995

[3]. Ivar Jacobson,Grady et al., The Unified Software Development Process, Addison Wesley, 1999

[4]. Rick Kazman, et al., "Requirements for Integrating Software Architecture and Reengineering Models:CORUM II, Reverse Engineering", 1998. Proceedings. 5th Working Conference , pp154-163

[5]. Gregory Abowd, e. al., "MORALE Mission Oriented Architectural Legacy Evolution Software Maintenance", 1997. Proceedings., International Conference on ,pp 150 -159, 1997

[6]. Chen X.P. et al., "Automatic Variable Classification for Cobol Program", In Proceedings of IEEE COMPSAC, 1994.

[7]. Joiner J.K., Tsai W.T., Chen X.P.,Subramanian S.,Boddu C. and Sun J., "Data-centered Program Understanding", In Proceedings of International Conference on Software Maintenance IEEE,pp272-282,Sept 1994.