

요구사항 스펙과 소스 코드 간 동기화를 위한 자동 프로젝트 문서 도구 개발

권하은^{1*}, 박보경^{2*}, 김영철^{3*}, 김영수^{4**}, 이상은^{5**}

*홍익대학교 컴퓨터정보통신공학과 소프트웨어공학연구실

** 정보통신산업진흥원 소프트웨어공학센터

e-mail : {¹kwon, ²park, ³bob}@somewhere.sck.ac.kr, {⁴ysgold, ⁵selee}@nipa.kr^{**}

Construction of Automatic Project Document Generator for synchronizing source code with requirement specifications

Haeun Kwon^{1*}, Bokyung Park^{2*}, R. Youngchul Kim^{3*}, Youngsoo Kim^{4**}, SangEun Lee^{5**}

*SE lab, Dept. of Computer Information Communication, Hongik University

**National IT Industry Promotion Agency

요약

현재 소프트웨어 요구사항은 전체 소프트웨어 개발 프로세스를 거쳐, 완성된 제품 인도 시 인수 기준으로 작용한다. 그러므로 이런 요구사항은 개발주기 전체에 걸친 매우 중요한 관리 수단이다[1]. 그러나 국내 다수의 중소기업은 이러한 문서화에 시간 및 비용이 부족하여, 개발 문서의 부재 혹은 코드와 문서가 불일치한 것이 현실이다. 이 문제를 해결하기 위해, 자동 프로젝트 문서 발생기(Automatic Project Document Generator)를 제안한다. 제안한 발생기를 통해 요구사항 스펙과 소스 코드 간 동기화로 요구사항 대로 개발이 진행되는지 확인 가능하다.

1. 서론

요구사항 명세가 설계에 반영되어 여러 개발 문서를 작성하는 기준이 된다. 또한 최종적으로 완성된 제품 인도 시에 인수기준으로 작용한다. 그러므로 요구사항은 개발주기 전체에 걸친 통제수단이라 할 수 있다[1].

그러나 국내 대다수의 중소기업의 경우 문서화에 투자할 시간 및 비용이 부족하다. 이로 인해 개발 문서가 부재하거나 개발 진척과 함께 문서가 갱신되지 않는다. 결과적으로 코드와 문서의 불일치가 발생하게 된다. 이를 해결하기 위해 자동 프로젝트 문서화를 제안한다. 이 도구(Automatic Project Document Generator)를 통해 요구사항 스펙과 소스 코드의 동기화가 가능하다. 앞으로는 요구사항 분석 단계에서 테스트 단계까지 소프트웨어 개발주기 전체에서 발생하는 문서를 자동화가 더욱 필요하다. 또한 기존 연구[2~4]와의 통합도 필요하다.

본 논문의 구성은 다음과 같다. 2 장에서 프로젝트 관리 소프트웨어를 소개한다. 3 장에서는 개발한 APDG 구조와 동작 과정을 설명한다. 마지막에서는 결론을 언급한다.

2. 프로젝트 관리 소프트웨어

프로젝트 관리는 프로젝트 성공을 위해 수행하는 행위를 의미하며, 프로젝트 관리 소프트웨어(PMS, Project Management Software)는 이를 보조해 주는 도구이다. 소프트웨어마다 제공하는 기능에 차이가 있으나 일반적으로 개발 문서 관리 및 형상관리 기능을 제공한다.



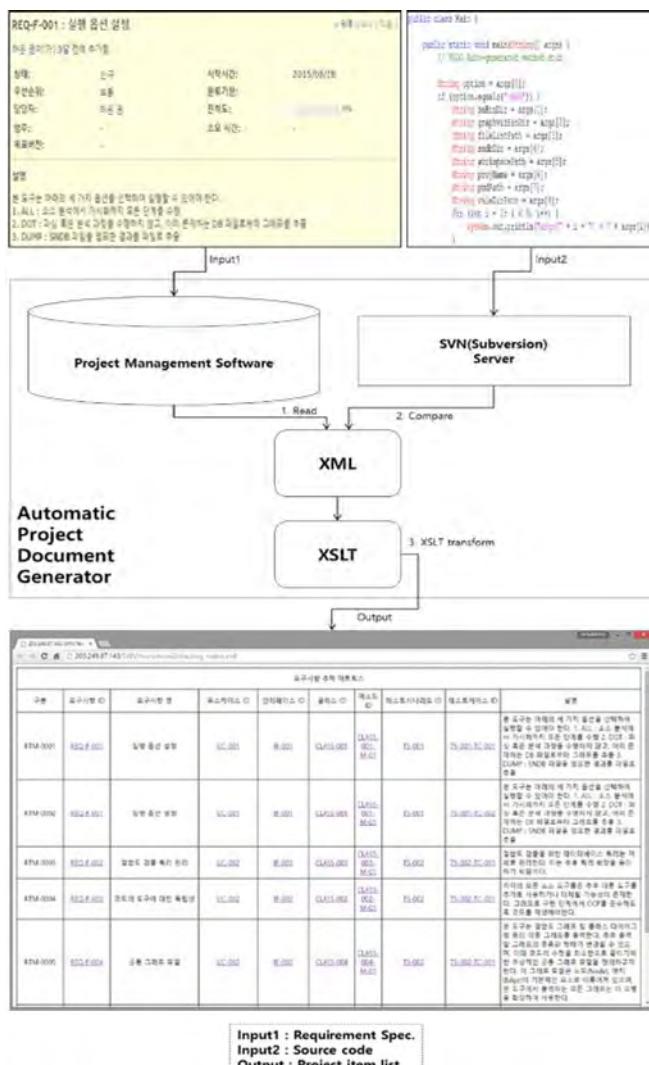
(그림 1) Redmine에 기록된 개발 문서

* 이 논문(저서)은 2015년 교육부와 한국연구재단의 지역혁신창의인력양성사업(NRF-2015H1C1A1035548)과 2015년도 정부(교육부)의 재원으로 한국연구재단의 지원을 받아 수행된 기초연구사업임(NRF-2013R1A1A2011601).

본 논문에서는 웹 기반의 공개 소프트웨어인 Redmine[5]을 사용한다. 이를 통해 요구사항 명세서, 유스케이스, 기능 명세서, 클래스와 메소드 명세서, 테스트 시나리오 및 테스트 케이스의 문서를 관리한다. 또한 각 문서에는 상위 단계 문서의 식별코드를 명시하여, 특정 문서가 어떤 상위 문서를 기반으로 작성되었는지 알 수 있도록 하였다. 예를 들어 그림 1에 나타난 UC-001 유스케이스는 REQ-F-001 요구사항 명세서를 기반으로 작성된 것을 의미한다.

3. Automatic Project Document Generator

APDG는 그림 2와 같이 구성된다. 입력으로는 요구사항 스펙과 소스 코드가 입력되는데, 각각 PMS와 SVN(Subversion) 서버에 저장된다. PMS에는 개발 문서가 저장되며, SVN 서버는 소스 코드의 형상관리를 수행하는 역할을 갖는다. 그리고 결과물로 프로젝트 항목 리스트(Project item list)를 출력한다. 이는 개발 문서간 추적성을 나타내는 문서로, 프로젝트 진행 단계에서 생성되는 문서 간에 연관성을 나열한 것이다. 자세한 동작 과정은 다음과 같다.



(그림 2) APDG 구성 및 동작 과정

먼저 PMS에 저장된 개발 문서들의 연관 관계를 분석한다. 2장에서 언급하였듯 모든 개발 문서는 상위 단계 문서에 대한 식별코드를 갖는다. APDG는 이 식별코드를 통해 요구사항 명세서를 시작으로 하는 일련의 문서간 관계를 추출하여, XML 데이터로 나타낸다. 예를 들어 그림 3은 REQ-F-001 요구사항으로부터 파생된 일련의 문서 목록을 나타낸 XML 데이터이다. XML 태그 중 'id'로 끝나는 태그는 문서의 식별 코드를 의미하고, 'link'로 끝나는 태그는 PMS에 존재하는 해당 문서의 URL을 나타낸다. 이는 최종적으로 프로젝트 항목 리스트에서 하이퍼링크로 각 문서에 접근할 수 있게 하기 위함이다.

```

<list>
  <task>
    <id>RTM-0001</id>
    <requirement_id>REQ-F-001</requirement_id>
    <requirement_link>http://...</requirement_link>
    <requirement_name>상행 옵션 설정</requirement_name>
    <usecase_id>UC-001</usecase_id>
    <usecase_link>http://...</usecase_link>
    <interface_id>IF-001</interface_id>
    <interface_link>http://...</interface_link>
    <class_id>CLASS-001</class_id>
    <class_link>http://...</class_link>
    <method_id>CLASS-001-M-01</method_id>
    <method_link>http://...</method_link>
    <test_scenario_id>TS-001</test_scenario_id>
    <test_scenario_link>http://...</test_scenario_link>
    <test_case_id>TS-001-TC-001</test_case_id>
    <test_case_link>http://...</test_case_link>
    <desc>본 도구는 아래의 세 가지 옵션을 선택하여 실행할 수 있어야 한다. 1. ALL : 소스 분석에서 기사화까지 모든 단계를 수행; 2. DOT : 파일 혹은 분석 과정을 수행하지 않고, 이미 존재하는 DB 파일로부터 그래프를 추출; 3. DUMP : SNDB 파일을 업그레이드한 결과를 파일로 추출</desc>
  </task>
</list>

```

(그림 3) 프로젝트 항목 리스트 XML 형식

다음으로 XML 데이터와 소스 코드를 비교한다. 소스 코드는 연관된 요구사항 식별 코드를 명시하여 SVN 서버에 전송되는데, APDG는 서버로부터 이를 읽어 들여, XML 데이터와 비교한다. 예를 들어 그림 4는 main.java 코드에 REQ-F-001 요구사항 식별 코드가 명시된 것이다. 이를 그림 3의 XML 데이터와 비교하는데, requirement_id 태그에 명시된 요구사항 식별 코드와 class_id 태그가 의미하는 소스 코드와 일치하는지를 비교한다. 비교 결과 둘 다 일치할 경우에는 XML 데이터를 파일로 출력한다.

```

// REQ-F-001
package selab.hongik.toolchain2;

import java.sql.Connection;
...
public class Main {
  public static void main(String[] args) {
    ...
}

```

(그림 4) 코드에 요구사항 식별코드 명시

마지막으로 생성된 XML 파일을 HTML 형식의 문서로 변환한다. 이 과정에는 XSLT(Extensible Stylesheet Language Transformation)이 사용된다. 이를 통해 그림 5와 같은 프로젝트 항목 리스트를 얻는다.

요구사항 추적 매트릭스									
구분	요구사항 ID	요구사항 명	유스케이스 ID	인터페이스 ID	클래스 ID	메소드 ID	테스트시나리오 ID	테스트케이스 ID	설명
RTM-0001	REQ-F-001	실행 옵션 설정	UC-001	IF-001	CLASS-001	CLASS-001-M-01	TS-001	TS-001-TC-001	본 도구는 아래의 세 가지 옵션을 선택하여 실행할 수 있어야 한다. 1. ALL : 소스 분석에서 가시화까지 모든 단계를 수행 2. DOT : 파싱 혹은 분석 과정을 수행하지 않고, 이미 존재하는 DB 파일로부터 그래프를 추출 3. DUMP : SNDB 파일을 엄포한 결과를 파일로 추출
RTM-0002	REQ-F-001	실행 옵션 설정	UC-001	IF-001	CLASS-001	CLASS-001-M-01	TS-001	TS-001-TC-002	본 도구는 아래의 세 가지 옵션을 선택하여 실행할 수 있어야 한다. 1. ALL : 소스 분석에서 가시화까지 모든 단계를 수행 2. DOT : 파싱 혹은 분석 과정을 수행하지 않고, 이미 존재하는 DB 파일로부터 그래프를 추출 3. DUMP : SNDB 파일을 엄포한 결과를 파일로 추출

(그림 5) 프로젝트 항목 리스트

4. 결론

본 논문의 목적은 요구사항 스펙과 소스 코드를 동기화 관점에서 자동 프로젝트 문서 발생 도구를 제안한다. 이를 위해 APDG를 개발하여 프로젝트 항목 리스트를 자동 생성하고, 개발 문서간 연관 관계를 분석한다. 이를 통해 문서 간 추적성 확보가 가능하다. 그 결과 개발 과정에서 요구사항대로 코드가 작성되는지 실시간으로 확인이 가능하다. 즉 요구사항 스펙과 소스 코드 간 동기화이다. 그러나 현재 APDG는 제한된 개발 문서만을 다룬다. 그러므로 향후에는 보다 다양한 개발 문서도 추가적으로 다루도록 한다. 또한 이번 연구와 기존 연구[2~4]와의 통합도 고려 중이다.

참고문헌

- [1] Liu. X., "A Quantitative Approach for Assessing the Priorities of Software Quality Requirements", Journal of systems and Software, Vol. 42, No. 8, 1998, pp.105-113
- [2] 권하은, 손현승, 서채연, 김영수, 박병호, 김영철, "기준 모듈 간의 결합도 및 응집도 개념과 객체지향 파라다임과의 관련 비교 연구", 한국정보과학회, 2014, pp.556-558
- [3] 권하은, 박보경, 이근상, 박용범, 김영수, 김영철, "코드 가시화부터 모델링 추출을 통한 역공학 적용", 한국정보처리학회, 제 21 권, 제 2 호, 2014, pp.650-653
- [4] Bokyung Park, Haeun Kwon, Young Soo Kim, R. Young Chul Kim, "Requirement Tracking Visualization for validating Requirement Satisfaction", The 5th International Conference on Convergence Technology 2015, Vol. 5, No. 1, 2015, pp.368-369
- [5] Lang, J., E. Davis, "Redmine-open source project management web-application.", <http://www.redmine.org/>, 2010