메뉴 구조의 필드간의 상호 연관관계를 기반으로 한 테스트 데이타 자동 생성 도구

(A Test Data Generation Tool based on Inter-Relation of Fields in the Menu Structure)

이 윤 정 † 최 병 주 ††
(Yoonjung Lee) (Byoungju Choi)

요 약 품질 인증 테스트는 소프트웨어의 품질을 결정하고 보증하기 위하여 인증 기관에서 제품 개발 후에 수행하는 테스트로써 해당 제품의 소스 코드 없이 제품 매뉴얼의 분석을 통하여 테스트가 이루어지는 경우가 대부분이다. 본 논문에서는 제품 매뉴얼에 기반한 테스트 데이타 생성을 위하여, 소프트웨어 패키지와 매뉴얼 분석 데이타로부터 테스트 데이타를 생성하는 것을 자동화한 '테스트 데이타 자동 생성 도구'(Manual-based Automatic Test data generating tool: *Ma*T)를 구현한다. *Ma*T의 입력 데이타는 소프트웨어 패키지와 매뉴얼의 분석 결과인데, 입력 데이타 구성을 위하여 '메뉴 기반 테스트 분석 모델'을 제안한다. 본 도구를 소프트웨어 패키지의 품질 인증 테스트에 적용함으로써 품질과 신뢰도가 향상된 소프트웨어 제품 개발에 기여할 수 있게 된다.

키워드 : 소프트웨어 패키지 테스트, 매뉴얼 기반 테스트, 블랙박스 테스트

Abstract The quality certification test is usually conducted by a certifying organization to determine and guarantee the quality of software after the software development phase, commonly without the actual source code, but with by going against the product' s manual. In this paper, we implement a Manual-based Automatic Test data generating tool: *Ma*T, the test technique based on manual, that automatizes producing the test data from analysis data of software package and manual. The input data of *Ma*T are the result of the analysis of software and manual. We propose 'menu-based test analysis model' in order to generate the input data. We believe that the proposed technique and tool help improving quality and reliability of the software.

Key words : software package test, manual-based test, black-box test

1. 서 론

소프트웨어 제품의 품질에 대한 관심이 점차 증가함에 따라 국제적 수준의 소프트웨어 품질 인증의 필요성도 증가하고 있다. 정부에서도 소프트웨어 품질 인증제도를 시행하여 일정 기준을 만족하는 제품에 품질 인증서 및 인증마크를 발급하고 있다. 이를 통하여 해당 제품의 신뢰성을 확보하고 판매를 촉진하는 동시에, 소프트웨어 개발 업체의 자발적인 품질 개선 노력을 유도하여 업체의 기술력을 향상시키고 수출 경쟁력을 강화하고자 한다. 그러나 소프트웨어 패키지의 품질 인증에 적용할 수 있는 테스트에 대한 기존 연구가 부족한 실정이다. 따라서 소프트웨어 패키지의 품질인증을 위한 체계적인 테스트 데이타 선정 기법이 요구된다.

소프트웨어의 품질 인증은 개발 인력을 인증하는 방법, 개발 프로세스를 인증하는 방법, 소프트웨어 제품을 인증하는 방법의 세 가지 관점으로 나눌 수 있다[1]. 개발 인력을 인증하는 방법은 전문적인 자격증 취득 시험을 통한 인증, 실제적인 프로젝트 경험 또는 학위 등이 있다[1]. 개발 프로세스를 인증하는 방법은 SW-CMM[2]이나 SPICE[2] 등이 있다. 제품을 인증하는 방법으로는 소프트웨어 제품의 품질을 평가하기 위한 국제 표준인

• 본 연구는 한국과학재단 목적기초연구(과제번호:2000-0-303-02-3) 지원으로 수행되었음.
† 비 회 원 : 이화여자대학교 컴퓨터학과
yoonjung@ewha.ac.kr
†† 종신회원 : 이화여자대학교 컴퓨터학과 교수
bjchoi@ewha.ac.kr
논문접수 : 2002년 3월 5일
심사완료 : 2002년 12월 31일

ISO/IEC 9126[3] 등을 적용할 수 있다.

소프트웨어 개발 인력의 인증이나 개발 프로세스의 인증만으로는 소프트웨어 제품의 품질을 인증하기는 어렵다[4,5]. 특히 개발 프로세스의 인증은 자체 개발 프로세스가 있는 대규모의 기업에 적용 가능하다. 하지만 품질 인증의 대상인 대다수의 소프트웨어 패키지는 개발 프로세스를 갖추고 있지 않은 소규모 기업들이 개발한 것이 대부분이다[4]. 다양한 분야에 속한 소프트웨어 패키지의 품질 인증 방법으로는 개발 프로세스 인증이나 개발 인력 인증보다는 소프트웨어 제품에 대한 품질 인증 방식을 적용하는 것이 적합하다. 따라서 본 논문에서는 제품 중심의 품질 인증에 적용할 수 있는 테스트 데이타 생성 도구의 설계 및 구현 방안을 제안한다.

소프트웨어 패키지의 제품 중심의 품질을 평가하는 데에는 ISO/IEC 9126[3]과 ISO/IEC 12119[6] 등의 국제 표준을 적용할 수 있다. 이들 표준에는 소프트웨어 품질 특성과 부특성, 그리고 그것을 평가하기 위한 메트릭이 정의되어 있으나, 각 메트릭을 측정하기 위한 테스트의 필요성에 대해서만 일부 기술되어 있을 뿐 구체적인 테스트 기법은 명시되어 있지 않다.

본 논문에서는 소프트웨어 패키지의 제품 매뉴얼과 패키지 실행 시 나타나는 메뉴를 분석한 데이타를 입력으로 하여 테스트 데이타를 자동으로 생성하는 '테스트 데이타 자동 생성 도구'(Manual-based Automatic Test data generating tool: *Ma*T)를 구현한다. 도구의 입력이 되는 메뉴 구조 분석에 적용하기 위하여 '메뉴 기반 테스트 분석 모델'을 제안한다. 또, *Ma*T가 입력 데이타로부터 테스트 데이타를 생성하기 위하여 '테스트 데이타 선정 기법'을 제안한다. 본 연구를 인증 테스트에 적용함으로써 소프트웨어 인증이 체계적으로 이루어 질 수 있으며, 연구결과를 품질과 신뢰도가 향상된 소프트웨어 제품을 개발하는데 활용할 수 있다.

본 논문은 2장의 관련 연구에 이어, 3장에서는 메뉴 구조 분석을 통한 입력 데이타 생성을 기술한다. 4장에서는 자동화 도구의 설계 및 구현에 대해서 기술하며, 5장에서는 본 도구에 대해 분석한다. 마지막으로 6장에서 결론 및 향후 연구 과제를 제시한다.

2. 관련 연구

2.1 컴비네토리얼 설계 기법

컴비네토리얼 설계 기법은 통계학에 의거해서 효과적인 실험을 구성하기 위해서 의학이나 산업 연구에 널리 사용되는 수학적인 구성 기법으로, 파라메터 사이의 인터렉션을 효과적으로 커버하기 위한 조합을 구성한다[7]. 컴비네토리얼 설계 기법이 기존의 직교 배열 기법을 적용하는 것보다 상대적으로 적은 수의 테스트 데이타를 생성하면서도, 에러 발견 측면에서 효율적인 테스트 데이타를 생성한다는 것이 여러 차례의 실험을 통해서 밝혀진 바 있다[8,9].

본 논문에서는 컴비네토리얼 설계 기법을 활용하여 소프트웨어 패키지의 매뉴얼 기반의 테스트를 위한 테스트 데이타를 생성하도록 한다.

2.2 테스트 데이타 생성 도구

시스템 테스트를 위한 도구로 인증 테스트에 적용 가능한 기능을 제공하는 것은 AETG와 TeamTest 등이 있다.

(1) AETG

텔코디아사(Telcodia Technologies)의 'AETG'(Automatic Efficient Testcase Generator)[7]는 웹 기반의 테스트 데이타 자동 생성 도구이다. AETG는 테스터가 시스템의 테스트 요구사항을 파라메터와 입력값으로 명시한 것을 입력으로 받아, 컴비네토리얼 설계 기법(combinatorial designs)[7]을 적용하여 입력 파라메터 값의 쌍을 이루는 조합(pair-wise combination)을 커버하는 최소의(minimal) 튜플(tuple)들로 구성된 테이블을 생성한다. 입력 파라메터의 조합은 각 파라메터에 대한 입력 값의 집합의 카티션 프로덕트(cartesian product)로부터 구성된다[9,10].

AETG는 테스터가 테스트 집합을 설계하는 것을 지원하는 도구로, 특히 입력값의 조합이 증가함에 따라 테스트 데이타가 기하급수적으로 증가하는 문제에 컴비네토리얼 설계 기법을 적용하여 효과적인 대안을 제시한다.

(2) TeamTest

Rational사의 TeamTest는 기능 테스트 및 성능 테스트 그리고 테스트 관리 기능을 제공하는 테스트 도구[11]이다. TeamTest는 현재 인증 기관에서 테스트 대상 소프트웨어 패키지의 기능을 자동으로 추출하는데 사용하고 있다.

TeamTest에는 인터넷용 어플리케이션의 기능 테스트, 분산 기능 테스트(Distributed functional test), 스모크 테스트(Smoke test)의 생성, 수정, 수행에 사용되는 테스트 자동화 도구인 Rational Robot이 포함되어 있다. TeamTest는 성능 테스트 시 성능상의 문제점을 식별하고, 반복 주기별 테스트를 지원하며, 웹사이트의 무결성을 관리하고, 어떤 기능 테스트 스크립트가 코드 변경의 영향을 받았는지를 식별할 수 있어 리그레션 테스트를 간편하게 수행할 수 있다. TeamTest는 소프트웨어의 개발 과정에서의 시스템 테스트에 유용한 도구

로, 스크립트 재사용등을 이용하여 매뉴얼 테스트를 위한 기능을 일부 제공한다[11].

3. 메뉴 구조 분석을 통한 입력 데이타 생성

품질 인증 테스트는 소프트웨어의 품질을 결정하고 보증하기 위하여 인증 기관에서 제품 개발 후에 수행하는 테스트이다[12]. 품질 인증 테스트는 소프트웨어 패키지의 소스 코드 없이 수행하게 되므로, 소프트웨어 제품의 매뉴얼에 대한 정확한 분석이 요구된다. 본 논문에서는 그림 1에서처럼 소프트웨어 패키지의 매뉴얼 분석 결과를 입력으로 받아 테스트 데이타를 생성하는 도구를 구현하였다. 입력 데이타 생성을 위하여 '메뉴 기반 테스트 분석 모델'을 적용한 프로그램의 분석방안을 제안하고, *MaT*가 입력 데이타로부터 테스트 데이타를 자동으로 생성하는데 적용하기 위하여 '테스트 데이타 선정 기법'을 제안한다.

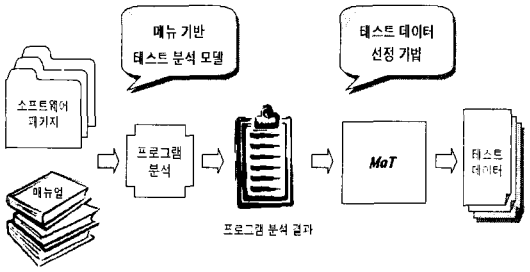

그림 1 매뉴얼 분석 데이타로부터 테스트 데이타 생성

메뉴 기반 테스트 분석 모델은 소프트웨어 제품 매뉴얼을 통하여 소프트웨어 패키지의 메뉴 구조를 분석하여 *MaT*의 입력 데이타로 이용하기 위하여 제안한 모델이다. 메뉴 기반 테스트 분석 모델을 기술하기 위하여 정의한 용어는 다음과 같다.

정의 1 : 스크린 S_i

프로그램의 메뉴 구조에서 최하위에 있는 스크린을 의미하는 것으로, 모든 스크린은 입력을 위한 필드를 갖는다고 가정한다. S_i는 메뉴 구조에서 i번째 스크린을 의미한다.

예 : 그림 2의 메뉴 구조에서 단말메뉴1은 S_1, 단말메뉴2는 S_2, 단말메뉴3은 S_3, 단말메뉴4는 S_4, 단말메뉴5는 S_5가 된다.

정의 2 : 필드 f_{ij}

스크린에서 값을 입력하는 부분을 의미한다. f_{ij}는 i번째 스크린의 j번째 필드를 의미한다.

예 : 그림 2에서 S_3의 입력필드3.1은 f_{31}, 입력필드3.2는 f_{32}, 입력필드 3.3은 f_{33}, 입력필드 3.4는 f_{34}, S_5의 입력필드 5.1은 f_{51}, 입력필드 5.2는 f_{52}가 된다.

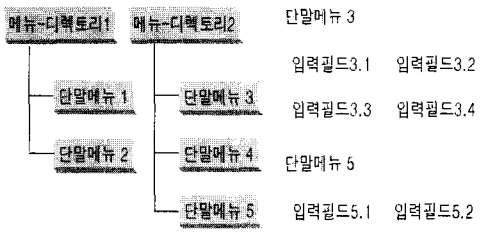

그림 2 메뉴 구조 예제

정의 3 : 시작 스크린(S_S), 종료 스크린(S_E)

시작 스크린은 S_S 라 하며 소프트웨어 패키지의 첫 번째 실행이 될 수 있는 스크린을 의미한다. 종료 스크린은 S_E 라 하며 마지막에 실행이 될 수 있는 스크린을 의미한다.

예 : 그림 2에서 S_S는 S_1, S_2, S_4, S_5로, S_E는 S_2로 가정한다.

정의 4 : 상호관계

필드와 필드 사이의 상호관계를 의미한다. '$f_{ip} \rightarrow f_{jq}$'라는 것은, 스크린 S_i의 필드 f_{ip}가 스크린 S_j의 필드 f_{jq}에 선행되어야 한다는 것을 의미한다. 다시 말해 스크린 S_i의 필드 f_{ip}의 입력 후에 스크린 S_j의 필드 f_{jq}를 입력하는 것이 의미가 있다는 것을 나타낸다.

예 : 그림 2에서 $f_{11} \rightarrow f_{32}$, $f_{32} \rightarrow f_{21}$, $f_{11} \rightarrow f_{51} \rightarrow f_{41}$, $f_{41} \rightarrow f_{21}$, $f_{11} \rightarrow f_{21}$의 상호관계를 갖는 것으로 가정한다.

정의 5 : 스크린 데이타(SD)

하나의 스크린을 테스트하기 위한 데이타를 의미한다. 즉 스크린의 각 필드를 테스트하기 위한 데이타이다.

정의 6 : 테스트 데이타(TD)

필드 사이의 상호관계에 따른 스크린 데이타의 시퀀스(a sequence of screen datum)를 의미한다. 즉, 소프트웨어 패키지의 TD는 S_S로부터 S_E까지의 SD들의 시퀀스로 구성된다.

정의 7 : 레벨

레벨은 S_S부터 S_E까지 스크린의 시퀀스에서의 길이로 정의한다. S_S 이면서 S_E가 되는 경우는 '레벨 =0'이다.

예 : 정의 4의 상호관계에 대한 예제에서 분석한 $f_{11} \rightarrow f_{21}$은 레벨 1에 해당하는 상호관계이다.

정의 8 : 확장 메뉴(EM)

테스트 대상 소프트웨어 패키지의 기능과 상호관계를

갖는 타 소프트웨어가 있을 경우, 타 소프트웨어의 기능들이나 대상 소프트웨어 패키지의 테스트 목적으로 필요한 추가 기능들, 예를 들면 'halt' 등의 기능을 의미한다.

정의 9 : 메뉴 기반 테스트 분석 모델

메뉴 기반 테스트 분석 모델은 위에서 정의한 스크린, 필드, 필드들의 상호관계와 각 필드의 valid와 invalid 입력 데이타들로 구성된다.

메뉴 기반 테스트 분석 모델을 적용한 프로그램의 분석 단계는 다음과 같다.

<메뉴 기반 테스트 분석 모델을 적용한 입력 데이타 구성>	
Step 1	제품 매뉴얼과 소프트웨어 패키지의 메뉴 구조로부터 테스트 대상 소프트웨어 분석
	Step 1.1 스크린, S_S와 S_E 분석
	Step 1.2 EM 정의
	Step 1.3 각 스크린의 중요도 점수화
Step 2	필드들 사이의 상호관계 분석
Step 3	각 스크린에 속한 각 필드에 대하여 valid와 invalid 입력 데이타를 결정

소프트웨어 패키지의 제품 매뉴얼과 소프트웨어 실행 시 나타나는 메뉴 구조를 분석하여 모든 스크린을 결정하고, 각 스크린을 분석하여 시작 및 종료 스크린이 될 수 있는 스크린들을 파악하여 S_S와 S_E를 결정한다. 테스트를 위하여 필요한 소프트웨어 패키지와 관련이 있는 외부 기능들이 있다면 이를 파악하여 EM을 결정한다. 테스트 대상 소프트웨어 패키지의 기능의 중요도를 반영한 테스트 데이타 선정을 지원하기 위해, 각 스크린의 중요도에 점수를 부과한다. Step 1.3은 테스트 데이타 선정 알고리즘을 적용하여 테스트 데이타를 생성 할 때, 필요 이상으로 많은 테스트 데이타를 선정하는 것을 방지하고, 중요한 기능을 중심으로 테스트 데이타를 선정하는데 사용하기 위함이다. 필드들 사이의 상호관계를 분석한 후, 각 필드에 대하여 valid와 invalid 데이타들을 선정한다. Valid 데이타 이외에도 invalid 데이타를 통해서, 예기치 못한 사용자의 사용에 대해 시스템의 반응을 테스트할 수 있다. *MaT*는 이러한 메뉴 기반 테스트 분석 모델을 적용한 메뉴 분석 결과를 입력으로 하여 테스트 데이타를 생성한다.

표 1과 표 2는 그림 2의 메뉴 구조 예제를 메뉴 기반 테스트 분석 모델을 적용하여 분석한 결과이다. 예를 들어 스크린 S_4는 S_S이고, 입력 필드 f_{41}을 갖고 있으며 valid 입력 값으로 'data'를, invalid 입력 값으로 'in_data'를 갖는다. 본 예에서는 모든 스크린이 같은 중요도를 갖고 있다고 가정한다.

표 1 메뉴 구조 예제의 메뉴 구조 분석 결과

스크린	S_1	S_2	S_3				S_4	S_5	
S_S	v	v	-				v	v	
S_E	-	v	-				-	-	
필드	f_{11}	f_{21}	f_{31}	f_{32}	f_{33}	f_{34}	f_{41}	f_{51}	f_{52}
Valid input	pn	y	y	50	fd	y	data	con	data1
	-	-	n	-	-	n	-	-	data2
Invalid input	-	-	-	999	in_fd	-	in_data	in_con	-

표 2 그림 2 메뉴 구조 예제의 상호관계 분석 결과

f_{11}	→	f_{32}		
f_{32}	→	f_{21}		
f_{11}	→	f_{51}	→	f_{41}
f_{41}	→	f_{21}		
f_{11}	→	f_{21}		

4. 테스트 데이타 자동 생성 도구: *MaT*

본 논문에서는 소프트웨어 패키지의 품질 인증을 위한 테스트 기법을 지원하기 위하여 자동화 도구인 *MaT*를 제안한다. *MaT*는 인증 테스트를 위한 테스트 데이타를 자동으로 생성하는 도구로, 입력으로 3장의 메뉴 기반 테스트 분석 모델을 적용하여 도출한 메뉴 분석 결과를 사용한다. 입력한 메뉴 분석 결과에 테스트 데이타 선정 기법을 적용하여 테스트 데이타를 자동으로 생성하여 Microsoft Excel 파일 형식으로 저장한다.

*MaT*는 Windows NT 5.0 환경에서 개발하였다. 개발 언어로는 Java2 SDK 1.2.2(JDK 1.2.2)를 이용하였고, 데이타베이스로는 JDBC를 사용하여 구현하였다. 그림 3은 *MaT*의 전체 구조를 UML의 패키지도로 나타낸 것이다. *MaT*는 데이타베이스와 '메뉴 분석 결과 업데이트 모듈'(Menu Structure Input Package), '상호관계 생성 모듈'(Relation Generator Package), '테스트

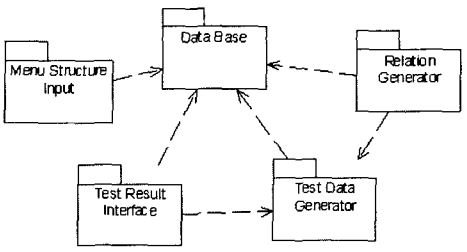

그림 3 패키지도 : *MaT*의 전체 구조

데이타 생성 모듈(Test Data Generator Package), '결과 인터페이스 모듈'(Test Result Interface Package)의 4가지 모듈로 구성된다.

그림 4는 테스트 데이타를 생성하기 위한 *MaT*의 사용 시나리오를 UML의 순서도로 표현한 것이다. 본 장에서는 *MaT*를 구성하는 각 모듈의 설계와 구현을 예제와 실행 화면과 함께 기술한다.

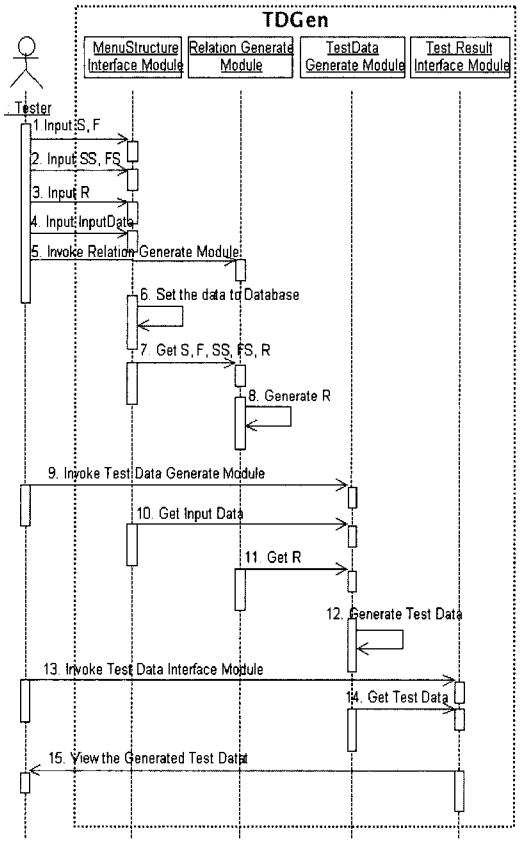

그림 4 *MaT* 사용 시나리오

<Alg 1>테스트 데이타 자동 생성 알고리즘	
Input	: 메뉴 기반 테스트 분석 모델을 적용한 테스트 대상 프로그램의 분석 결과 (스크린, 필드, S_S, S_E, 상호관계, Input Data)
Output	: 테스트 데이타 (*SD*, *TD*)
Step	:
1	메뉴 분석 결과를 데이타베이스에 저장
2	Ss부터 SF까지 상호관계 생성
3	테스트 데이타 생성
3.1	'필드 데이타 조합 알고리즘'을 적용하여 *SD* 생성
3.2	'테스트 데이타 생성 알고리즘'을 적용하여 *TD* 생성

4.1 메뉴 분석 결과 업데이트 모듈

*MaT*에서 테스트 데이타를 자동으로 생성하기 위하여, 메뉴 기반 테스트 분석 모델을 적용한 프로그램 분석 결과가 입력 값으로 요구된다. 메뉴 분석 결과 업데이트 모듈은 사용자로부터 소프트웨어 패키지의 분석 결과를 입력 받아 테스트 대상 프로그램의 메뉴 구조를 데이타베이스에 저장하는 모듈이다.

사용자는 *MaT*에서 지원하는 트리 구조로 메뉴 분석 결과인 스크린, 필드, S_S, S_E, 상호관계, valid와 invalid 입력 데이타 등을 입력한다. 본 논문에서는 3장의 표 1, 표 2의 예제를 입력 데이타로 사용한다.

그림 5는 *MaT*의 시작 화면으로, 사용자로부터 프로젝트 제목과 간단한 내용을 입력받는다.

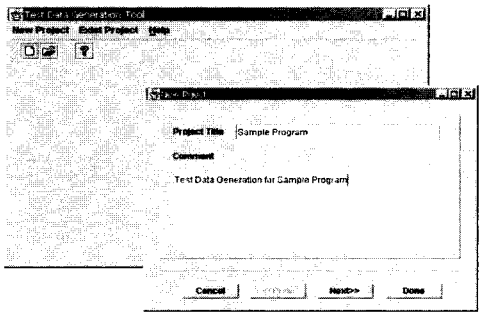

그림 5 *MaT* 시작 화면

그림 6, 그림 7, 그림 8은 메뉴 분석 결과를 입력한 화면을 보여 준다. 사용자는 분석한 프로그램의 메뉴를 트리 구조로 입력 할 수 있으며, S_S, S_E와 필드간의 상호관계는 서로 관련된 필드들을 마우스로 클릭하여 입력할 수도 있다. 각 스크린의 중요도는 '스크린 이름:중요도'의 형태로 입력한다.

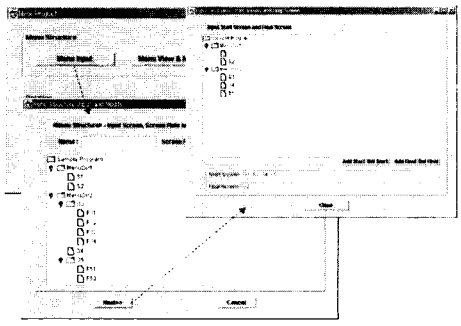

그림 6 스크린, 필드, S_S, S_E 입력 화면

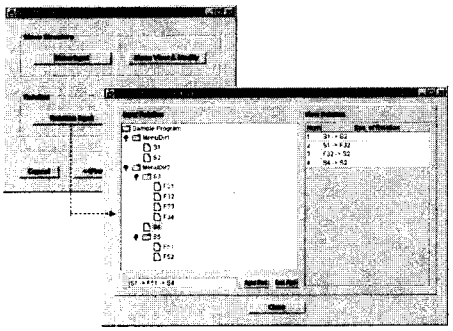

그림 7 상호관계 입력 화면

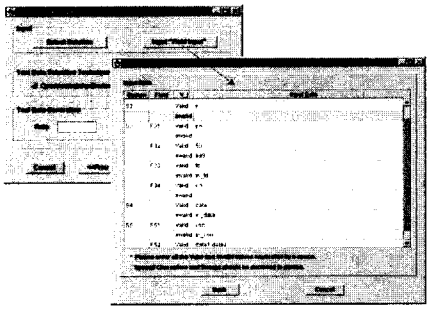

그림 8 입력 데이타 입력 화면

4.2 상호관계 생성 모듈

*Ma*T에서 생성하는 *TD*는 S_S부터 S_E까지의 시퀀스로 구성되어야 하기 때문에, 상호관계 생성 모듈은 데이타베이스에 저장되어 있는 스크린, 필드, S_S, S_E, 상호관계에 관한 정보를 이용하여, S_S부터 S_E까지의 상호관계를 생성한다. 이는 사용자가 상호관계를 분석할 때 S_S부터 S_E까지의 시퀀스를 고려하지 않아도, 본 논문에서 구현한 도구에서 자동으로 상호관계를 생성하여 사용자가 보다 편리하게 도구를 사용하도록 지원하기 위함이다.

메뉴 구조의 입력이 끝나면 *Ma*T는 S_S부터 S_E까지 시퀀스를 자동으로 생성한다. 표 3은 표 1과 표 2의 메뉴 분석 결과를 입력으로 *Ma*T에서 생성한 S_S부터 S_E까지의 상호관계를 나타낸 것이고, 그림 9는 *Ma*T에서 상호관계를 생성하는 단계를 보여준다.

4.3 테스트 데이타 생성 모듈

본 논문에서는 MaT가 소프트웨어 패키지의 매뉴얼에 기반한 테스트 데이타를 자동으로 생성하도록 하기 위하여 '테스트 데이타 선정 기법'으로서 '필드 데이타 조합 알고리즘'(Field Data Combination Algorithm)과 '테스트 데이타 생성 알고리즘'(Test Data GenerationAlgorithm)

표 3 *Ma*T에서 생성한 S_S부터 S_E까지 상호관계

레벨	생성한 상호관계
0	f_{21}
1	f_{11} → f_{21}
1	f_{41} → f_{21}
2	f_{11} → f_{32} → f_{21}
3	f_{11} → f_{51} → f_{41} → f_{21}

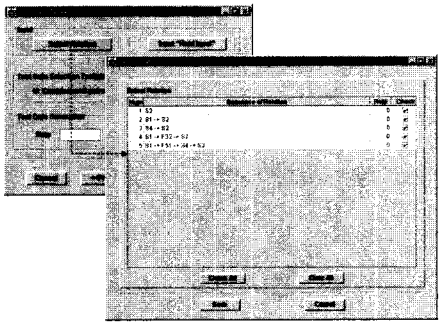

그림 9 상호관계 생성 단계

의 두 가지를 제안한다. 필드 데이타 조합 알고리즘은 컴비네토리얼 설계 기법[7, 8]을 활용하여 필드의 입력 값으로부터 *SD*를 생성하는 단계와 상호관계를 갖는 필드들의 입력 데이타로부터 필드 입력 데이타의 조합을 생성하는 단계에 적용하는 알고리즘이다. 테스트 데이타 생성 알고리즘은 필드 데이타 조합 알고리즘을 적용하여 생성한 상호관계를 갖는 필드들의 조합과 생성한 *SD*로부터 *TD*를 생성하는 알고리즘이다.

<*Ma*T의 테스트 데이타 생성 절차>
• 각 스크린별로 *SD* 생성
• 상호관계를 갖는 필드들의 조합 데이타 생성
• 임계값 이상의 상호관계에 대하여, 각 상호관계별로 *TD* 생성

테스트 데이타 생성 모듈은 데이타베이스에 저장되어 있는 입력 데이타와 *Ma*T에서 생성한 상호관계를 입력으로, '필드 데이타 조합 알고리즘'과 '테스트 데이타 생성 알고리즘'을 이용하여 테스트 데이타를 생성한다. 중요한 기능에 대한 테스트만 요구될 때, 혹은 테스트 데이타가 필요 이상으로 많이 생성되는 것을 방지하기 위

해서 그림 9의 Rate 입력란에 임계값을 입력하면, 테스트 데이타 생성 모듈은 상호관계를 이루는 스크린의 중요도의 합이 임계값 이상인 TD만 생성한다. 생성한 테스트 데이타를 데이타베이스에 저장하고 MicroSoft Excel 파일 형식으로 파일로도 저장한다.

테스트 데이타 생성 모듈에서 SD와 상호관계를 갖는 필드의 조합을 생성하는데 적용하는 필드 데이타 조합 알고리즘은 다음과 같다.

<Alg 2> 필드 데이타 조합 알고리즘

Input:	필드들의 상호관계, 각 필드의 valid와 invalid 입력 데이타
Output:	상호관계를 갖는 필드의 입력 데이타를 조합한 시퀀스
Step 1	상호관계를 갖는 필드의 입력 데이타들로 후보 집합과 연관 집합 구성
Step 1.1	상호관계를 갖는 모든 필드의 입력 데이타를 조합하여 후보집합을 생성
Step 1.2	모든 필드에 대해, 2개씩 pair를 구성하여 입력 데이타 조합의 연관 집합을 생성
Step 2	후보 집합에서 임의의 집합을 첫번째로 선택
Step 2.1	이 집합에서 커버(cover)한 pair를 연관 집합에서 삭제
Step 3	남은 후보 집합에서 다음 집합을 선택
Step 3.1	남은 각 후보 집합에서 커버한 pair의 수 계산, 카운트에 저장
Step 3.2	카운트가 가장 높은 집합 선택
Step 3.3	이 집합에서 커버된 pair를 연관 집합에서 삭제
Step 4	연관 집합의 모든 집합이 커버될 때 까지 Step 3 반복
Step 5	선택된 집합들로 시퀀스 구성

표 4 MaT에서 생성한 메뉴 구조 예제의 SD

스크린	S_1	S_2	S_3				S_4	S_5	
필드	f_{11}	f_{21}	f_{31}	f_{32}	f_{33}	f_{34}	f_{11}	f_{51}	f_{52}
Valid	pn	y	y	50	fd	y	data	con	data1
	"	"	n	50	fd	n	"	con	data2
	"	"	y	50	fd	n	"	"	
	"	"	n	50	fd	y	"	"	
Invalid			y	999	fd	n	in_data	in_con	data1
	"	"	n	50	in_fd	y	"	"	

필드 데이타 조합 알고리즘은 테스트 케이스의 수를 최소로 하면서도 효율적인 조합을 생성하는 방안에 관한 기존연구[7,9,10]를 바탕으로 테스트 데이타 조합의 pair-wise를 커버하는 데이타를 생성하는 컴비네토리얼 설계 기법[7]을 이용하여 구현하였다. 표 4는 MaT에서 생성한 SD를 나타낸 것이다. 예를 들어, 스크린 S_5는 valid 테스트 데이타로 (con, data1), (con, data2)의 2가지를, invalid 테스트 데이타로 (in_con, data1)의 1가지를 갖는다.

제안하는 테스트 데이타 생성 알고리즘은 다음과 같다.

<Alg 3> 테스트 데이타 생성 알고리즘

Input:	각 스크린의 SD, 상호관계
Output:	레벨별 TD
Step 1	레벨 = 0 (S_S = S_E) TD = {해당 스크린의 SD}
Step 2	레벨 ≥ 1 (S_S ≠ S_E) 1) 상호관계를 갖는 필드 데이타 조합 생성 2) 1)에서 생성한 조합에 포함된 스크린의 SD를 연결 3) TD = {2)에서 연결한 SD}

TD의 생성은 필드간의 상호관계의 경로(path)를 레벨로, 각 레벨별로 단계적으로 이루어진다. '레벨 = 0', 즉 해당 스크린이 S_S이면서 S_E인 경우의 TD는 Step 1에서 생성한 스크린의 SD이다. 레벨이 1 이상인 경우에는 해당 레벨에서 1) 필드 데이타 조합 알고리즘을 적용하여 상호관계를 갖는 필드들의 필드 데이타 조합을 생성한다. 2) 1)에서 생성한 조합과 일치하도록 상호관계를 갖는 필드를 포함하는 스크린의 SD를 연결한다. 3) 레벨이 1 이상인 경우의 TD는 2)에서 연결한 테스트 데이타이다.

표 5는 MaT에서 생성한 TD의 일부를 나타낸 것이

표 5 상호관계별 테스트 데이타 TD (일부)

$f_{11} \rightarrow f_{32} \rightarrow f_{21}$

	f_{11}	f_{31}	f_{32}	f_{33}	f_{34}	f_{21}
Valid	pn	y	50	fd	y	y
Invalid	pn	y	999	fd	n	y

$f_{11} \rightarrow f_{51} \rightarrow f_{41} \rightarrow f_{21}$

	f_{11}	f_{51}	f_{51}	f_{41}	f_{21}
Valid	pn	con	data1	data	y
Invalid	pn	in_con	data1	data	y
	pn	con	data2	in_data	y

다. 예를 들어 표 3의 레벨 3의 '$f_{11} \rightarrow f_{31} \rightarrow f_{41} \rightarrow f_{21}$' 상호관계의 경우, 상호관계를 갖는 필드 데이타 조합이 valid 조합으로 (pn, con, data, y)의 1가지를 생성하였고 invalid 조합으로 (pn, in_con, data, y), (pn, con, in_data, y)의 2가지를 생성하였다. 이 상호관계를 구성하는 각 스크린의 *SD*를 연결한 결과는 표 5에서 볼 수 있는 것처럼 (pn, con, data1, data, y)의 valid *TD*를 생성하였고, (pn, in_con, data1, data, y), (pn, con, data2, in_data, y)의 invalid *TD*를 생성하였다.

4.4 결과 인터페이스 모듈

결과 인터페이스 모듈은 Excel 파일로 저장한 테스트 데이타 집합과 테스트 데이타 시퀀스를 보여준다. 또, 생성된 테스트 데이타 시퀀스를 가지고 테스트를 수행한 결과를 사용자가 입력하면 그 결과를 데이타베이스에 저장한다.

그림 10과 그림 11은 테스트 데이타 생성 결과를 보여 주는 화면이다. 4.3절의 표 4, 표 5의 *SD*와 *TD*가

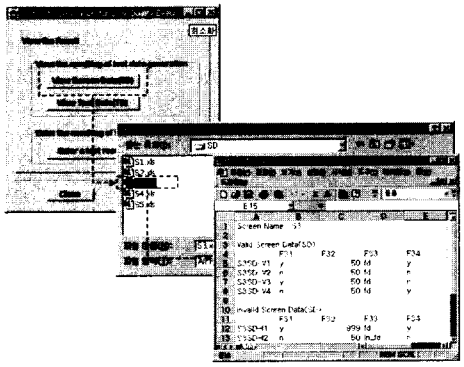

그림 10 *SD* 생성 결과

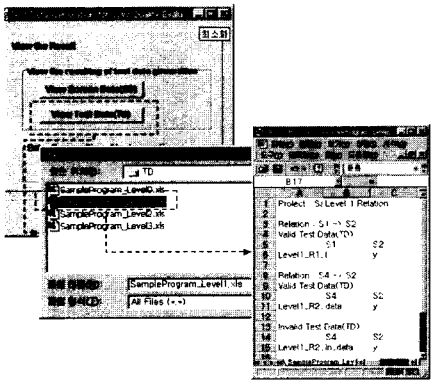

그림 11 *TD* 생성 결과

각각 해당 프로젝트의 이름을 갖는 디렉토리 밑의 'SD' 디렉토리와 'TD' 디렉토리에 Microsoft Excel 파일로 저장된다. *SD*는 각 스크린별로 파일이 생성되며 *TD*는 각 레벨별로 파일이 생성된다.

5. *MaT* 분석

본 논문에서 제안한 *Ma*T를 테스트 데이타 생성 도구인 AETG, 현재 소프트웨어의 기능 분석에 사용하는 TeamTest와 "매뉴얼로부터의 입력 데이타 생성 방안 제공성", "매뉴얼/소프트웨어 분석 자동화 지원성", "매뉴얼 기반의 테스트 데이타 자동 생성", "기타 특징" 등의 관점에서 비교한 결과는 표 6과 같다. 이에 대하여 구체적으로 살펴보면 다음과 같다.

(1) 매뉴얼로부터의 입력 데이타 생성 방안 제공성

본 논문에서 제안하는 *Ma*T는 도구에서 자동으로 매뉴얼 분석을 수행하지 않지만, *Ma*T의 사용 매뉴얼에서 입력 데이타 생성을 위한 테스트 대상 소프트웨어 매뉴얼로부터의 입력 데이타 생성 방안을 제공한다. AETG는 프로그램을 분석한 결과로부터 입력 데이타를 생성하는 방안을 매뉴얼을 통하여 제공하고, 그것을 매뉴얼로부터 입력 데이타를 생성하는 데 응용할 수는 있다. 그러나 TeamTest는 테스트를 위하여 매뉴얼로부터의 입력 데이타 생성에 대한 방안을 제공하지는 않는다.

(2) 매뉴얼/소프트웨어 분석 자동화 지원성

*Ma*T는 사용자가 입력한 필드간의 상호관계로부터 소프트웨어의 시작부터 종료까지의 필드간의 시퀀스를 자동으로 생성하여 주기 때문에 매뉴얼/소프트웨어 분석의 자동화를 일부 지원한다. TeamTest의 경우에는 소프트웨어 제품의 메뉴 구조를 자동으로 분석하여 주는 기능을 지원한다. AETG는 이러한 기능을 지원하지 않는다.

(3) 매뉴얼 기반의 테스트 데이타 자동 생성

*Ma*T는 소프트웨어 패키지와 매뉴얼을 분석한 사용자의 입력 데이타로부터 테스트 데이타를 자동으로 생성하여 준다. AETG도 입력 데이타로부터 테스트 데이타를 자동으로 생성하지만, 매뉴얼 기반의 테스트 데이타 자동 생성이 아니고 그 기법을 매뉴얼 기반의 테스트에 응용할 수 있을 뿐이다. TeamTest의 경우 매뉴얼 기반의 테스트를 위하여 테스트 스크립트의 사용을 지원하여 반복적인 테스트를 지원하나 매뉴얼 기반의 테스트 데이타를 자동으로 생성하지는 않는다.

(4) 기타 특징

*Ma*T는 3장의 메뉴 기반 테스트 분석 모델을 적용하

표 6 *Ma*T와 다른 도구와의 비교 분석

특징 \ 도구	AETG	TeamTest	***Ma*T**
매뉴얼로부터의 입력데이타 생성 방안 제공성	일부 제공	제공하지 않음	제공
매뉴얼/소프트웨어 분석 자동화 지원성	지원하지 않음	지원	일부 지원
매뉴얼 기반의 테스트 데이타 자동 생성	△	X	O
기타 특징	웹 기반의 테스트 데이타 생성 도구	기능테스트, 성능테스트, 테스트 관리 기능 제공	매뉴얼 기반 테스트를 위한 테스트 데이타 생성 도구

여 소프트웨어 패키지를 분석한 입력 데이타를 가지고 1)프로그램의 시작부터 종료까지의 시퀀스를 생성하는 테스트 대상 소프트웨어 분석을 통하여 2)매뉴얼 기반의 테스트를 위한 테스트 데이타를 자동으로 생성한다. AETG는 웹 기반의 테스트 데이타 생성 도구로 사용자가 입력 파라메터와 그 입력값, 그리고 릴레이션을 입력하면 테스트 데이타를 생성한다. TeamTest는 인증 테스트에서 소프트웨어의 기능을 자동으로 추출하는데 사용하는 도구이다.

*Ma*T는 위의 기타 특징에서 기술한 것처럼 테스트 데이타 생성 도구이다. *Ma*T로부터 생성한 테스트 데이타를 통해 검출할 수 있는 소프트웨어 패키지의 오류는 다음과 같다. 오류의 종류는 테스팅에 관한 기존 문헌[13]을 참조하여 분류하였다. 메뉴 기반 테스트 분석 모델을 통한 프로그램의 분석으로 기존 테스트에서 간과하기 쉬운 invalid 입력을 탐지(detect)하지 못하는 오류, 기능 사이의 작동순서(sequence)에 따라서 발생하는 오류, 타 소프트웨어와 상호작용(interaction)에서 발생하는 오류를 감지 할 수 있다. 이는 제안하는 방안이 기존의 수동적(manunal)인 테스트에서 간과하였던 invalid 입력 데이타 값을 이용한 테스트 데이타의 생성과 프로그램의 각 기능들의 상호관계를 레벨별로 정의하여 테스트 데이타를 생성하기 때문이다.

- 기능의 오류
  매뉴얼에 나타난 개별 기능이 제대로 구현되어 있지 않은 경우의 오류로, 레벨 0에서의 TD를 테스트하여 검출 할 수 있다.
- Invalid 입력을 탐지(detect)하지 못하는 오류
  Invalid 입력을 인식하여 그에 해당하는 반응(behavior)을 하지 못하는 경우의 오류로, invalid 입력 데이타 값을 입력 데이타로 선정하여 검출할 수 있다.
- 기능 사이의 작동순서(sequence)에 따라서 발생하는 오류
  테스트 대상 패키지 소프트웨어의 기능이 서로 상호작용(interaction)을 하면서 발생하는 경우의 오류이다. 이러한 오류는 발견하기가 어려운데 에러를 찾기 위해 테스트 케이스를 증가시키다 보면 테스트 케이스의 수가 많아지는 문제점 - 증가된 테스트 케이스 수에 비해 에러 발견측면에서의 효율성이 낮은 문제점 - 있다. 이러한 오류는 서로 상호작용을 하는 기능의 상호관계를 파악하여 테스트 데이타 선정에 추가함으로써 검출 할 수 있다.
- 타 소프트웨어와 상호작용(interaction)에서 발생하는 오류
  테스트 대상 소프트웨어 패키지의 기능이 타 소프트웨어의 기능과 서로 상호작용을 하면서 발생하는 오류이다. 이러한 오류는 타 소프트웨어의 기능을 *EM*으로 분석하여 필드간의 상호관계에 포함하여 *TD*를 생성하여 테스트함으로써 검출 할 수 있다.

6. 결론 및 향후 연구 과제

본 논문은 블랙박스의 특성을 갖는 소프트웨어 패키지의 매뉴얼 기반의 테스트를 위한 테스트 데이터 생성 도구를 제안하였다. 소프트웨어 패키지의 효과적인 테스트를 위해서는 정확한 분석을 통한 입력이 요구되기 때문에, 이를 지원하기 위하여 '메뉴 구조 분석 모델'을 제안하였다. 입력으로부터 테스트 데이타를 자동으로 생성하기 위하여 '테스트 데이타 자동 생성 알고리즘'을 제안하였다. *Ma*T는 메뉴 분석 결과를 입력으로 하여 상호관계의 중요도가 사용자가 입력한 임계값 이상인 테스트 데이타를 생성한다. 즉, 불필요하게 많은 테스트 데이타를 생성하는 것을 방지하면서 중요한 기능 위주의 테스트 수행을 지원한다. 생성한 테스트 데이타가 데이타베이스에 저장되기 때문에 리그레션 테스트 등을 위해서 재사용 할 수 있다. 본 논문에서 제안한 *Ma*T를 이용한다면 보다 신뢰성 있는 매뉴얼 기반의 테스트가 수행될 수 있고 소프트웨어 패키지의 품질 향상을 위한 테스트가 가능할 수 있을 것이다. 제안하는 도구를 패키지 소프트웨어 품질 인증 테스트에 적용하면 매뉴얼 기반의 인증 테스트가 보다 체계적으로 수행될 수 있다.

향후 연구 과제로는 소프트웨어 패키지가 속한 개발

도메인의 특성을 고려한 테스트 기법에 대한 연구가 요구된다. 그리고, '품질 인증 테스트의 자동화'를 위하여 생성한 테스트 데이타로 테스트를 자동으로 수행하는 방안에 대한 연구가 요구된다.

참 고 문 헌

[1] J. Voas, "The Software Quality Certification Triangle," Crosstalk, Vol. 11, No. 11, pp.12-14, Nov. 1998.

[2] Roger S. Pressman, "Software Engineering: A Practitioner's Approach, 5E," McGraw-Hill, 2000.

[3] ISO/IEC 9126: Information Technology - Software Product Evaluation - Quality Characteristics and Guidelines for Their Use, ISO, 1991.

[4] Jeffrey M. Voas, "Developing Usage-Based Software Certification Process," IEEE Computer, Vol.33, No. 8, 2000.

[5] William T. Council, "Third-Party Testing and the Quality of Software Components," IEEE Software, Vol. 16, No. 4, pp 55-57, 1999.

[6] ISO/IEC 12119 Information technology - Software packages - Quality requirements and testing.

[7] M.Hall Jr., "Combinatorial Theory," Willey Interscience, New York, 1986comp.

[8] Cohen, D. M., Dalal, S. R., Parelius, Jesse. and Patton, G. C., "The Combinatorial Design Approach to Automatic Test Generation," Seventh International Symposium on Software Reliability Engineering (ISSRE). White Plains, NY, Oct. 30 to Nov. 2, 1996.

[9] Cohen, D. M., Dalal, S. R., Fredman, M. L., and Patton, G. C., "The AETG System: An Approach to Testing Based on Combinatorial Design," IEEE Transaction on Software Engineering. Volume 23, Number 27, July 1997.

[10] Dalal, S. R., Jain, A., Karunanithi, N., Leaton, J. M., and Lott, C. M., "Model Based Testing of a Highly Programmable System," Proceedings of International Symposium on Software Reliability Engineering (ISSRE) - 1998, November 1998, Paderborn, Germany.

[11] Rational Software, "The Key to Successful Testing: Test Planning": White Paper, Rational Software Corporation.

[12] Jeffrey Voas, "Third-Party Usage Profiling: A Model for Optimizing the Mass-Marketed Software Industry," IEEE Software, 2000.7.

[13] William C. Hetzel, "The Complete Guide to Software Testing, Second edition," QED Information Services INC. 1988.

이 윤 정

1992년~1996년 이화여대 수학과 학사
1999년~2002년 이화여대 컴퓨터학과 석사
2002년~현재 LG CNS 소프트웨어 공학 센터 관심분야는 소프트웨어 공학, 소프트웨어 테스트, Product-line Engineering

최 병 주

1979년~1983년 이화여대 수학과 학사 1984년~1985년 Purdue Univ. Computer Science 학사수료. 1986년~1987년 Purdue Univ. Computer Science 석사 1987년~1990년 Purdue Univ. Computer Science 박사. 1991년~1992년 삼성종합기술원 1992년~1995년 용인대 전산통계학과 조교수. 1995년~현재 이화여대 컴퓨터학과 부교수. 관심분야는 소프트웨어공학, 소프트웨어 테스트, 소프트웨어 및 데이타 품질 측정