테스트 프로세스 수행 도구
(Test Process Execution Tool: Test PET)

천 은 정 † 최 병 주 ††
(Eunjung Chun) (Byoungju Choi)

요 약 개발 방법론과 도메인의 특성을 반영하여 테스팅 하기 위해서는 프로세스 표준을 테일러링 해야 하며 테일러링 된 프로세스에 따라 테스트를 수행해야 한다. 그러나 상용화된 테스트 도구들은 테일러링 된 프로세스의 일부만을 지원하기 때문에 실제 테스팅 현장에서는 개발 환경에 적합한 테스트 도구를 구입하거나 개발해야 한다.

본 논문에서는 프로덕트 라인 개발 개념을 이용한 테스트 프로세스 수행 도구 개발 방안을 제안하고 이를 '테스트 프로세스 수행 도구(Test Process Execution Tool: Test PET)'로 구현한다. 표준에 명시된 테스트 활동의 공통점과 차이점을 추출하여 이를 컴포넌트로 명세 및 구현하고 개발 방법론과 도메인에 맞는 특성을 반영하여 특정 개발 방법론과 도메인에 맞는 테스트 수행 도구를 개발하는 단계를 제안한다. 제안한 방안으로 구현된 Test PET은 개발 방법론과 도메인, 어플리케이션에 맞는 테스트 프로세스를 생성 한 후 생성된 테스트 프로세스에 따라 테스트를 수행할 수 있는 도구이다.

키워드 : 테스트 프로세스, 테스트 자동화, 테스트 도구, 프로덕트 라인

Abstract In order to test reflecting the features of a development methodology and domain, it is required to tailor process standards and perform testing according to the tailored process. However, since commercial testing tools support only a part of the tailored process, it is essential to either acquire or develop testing tools appropriate for a development environment.

This paper proposes a method to develop a test process execution tool which has common features of standards, and variousness in methodologies and domains. 'Test Process Execution Tool: Test PET' which is a test process execution tool developed adapting the concept of product line. Our Test PET generates the test process suitable for the development methodology and domain and then executes the produced test process.

Key words : Test Process, Test Automation, Testing Tool, Product Line

1. 서 론

테스트 도구의 사용으로 테스팅 작업과 제품 개발 기간을 단축할 수 있으며, 제품 품질을 향상시킬 수 있다. 그러나 테스트 프로세스 전체를 지원하는 테스트 도구는 많지 않으며, 전체 테스트 프로세스를 지원하는 도구라도 제공하는 기법이 한정적이고 확장이 어렵다는 단점이 있다. 테스트 프로세스 각 단계에 적합한 기능을 가진 테스트 도구를 선택하여 사용해야 한다. 전체 테스트 프로세스를 지원하면서 각 프로세스 단계에 맞는 다양한 테스트 기법을 제공할 수 있는 테스트 도구가 필요하다.

특정 도메인에 속하는 다양한 어플리케이션의 공통점과 차이점을 추출하여 이를 재사용 함으로써 고품질의 소프트웨어를 빠른 시간 내에 시장에 출시하고자 프로덕트 라인 개념[1]이 등장하였다. 프로덕트 라인(product line)은 선택된 특정 마켓의 요구를 만족하면서 공통적으로 다룰 수 있는 특성을 공유하는 제품(product)들의 집합이다[1]. 소프트웨어 프로덕트 라인에는 프로덕트와 프로덕트 라인 관계, 프로덕트의 제품간의 관계 등 다양한 관계가 존재한다[1,2]. 다양한 관계를 반영하여 테스트를 수행해야 하기 때문에 테스팅 작업과 테스트 산출물의 재사용이 매우 중요하다. 그러나 특정 도메인의 요구에 맞는 테스팅 작업과 테스트 산출물을 생성 및 관리하는 도구를 구입하거나 개발하는 것이 용이하

· 본 연구는 한국과학재단 목적기초연구(과제번호:2003-000-10139-0)의 부분지원 및 대학 IT연구센터 육성 지원사업의 부분지원으로 수행되었음
† 비 회 원 : 이화여자대학교 컴퓨터학과
myring@ewha.ac.kr
†† 종신회원 : 이화여자대학교 컴퓨터학과 교수
bjchoi@ewha.ac.kr
논문접수 : 2003년 2월 3일
심사완료 : 2003년 11월 26일

지 않다.

본 논문에서는 프로덕트 라인 개발 개념과 프로세스를 적용한 테스트 프로세스 수행 도구 개발 방안에 대해 제안하고 '테스트 프로세스 수행 도구'(**Test Process Execution Tool: Test PET**)를 개발한다. 프로덕트 라인은 코어 어셋 개발 프로세스와 개발된 코어 어셋으로 제품을 개발하는 제품 개발 프로세스로 이루어진다. 코어 어셋 개발 단계에서는 도메인의 공통 요구 사항들을 도출하여 추상화 시킨 후 도메인 내의 여러 어플리케이션과의 공통점과 차이점을 분류, 분석을 통하여 도메인을 위한 코어 어셋을 컴포넌트로 구현한다. 이 단계에서는 프로덕트 라인의 범위, 코어 어셋과 생산 계획(production plan)을 정의한다. 제품 개발 단계에서는 생산 계획에 따라 코어 어셋들을 맞춤 하여 특정 제품을 개발한다. 즉 코어 어셋 개발 프로세스의 결과물인 프로덕트 라인 범위, 코어 어셋, 생산 계획과 각 제품의 요구사항을 입력으로 반복적인 프로세스를 통해 제품을 생산한다[1]. 따라서 본 논문에서 제안한 테스트 프로세스 수행 도구의 개발 단계는 프로덕트 라인 개발 프로세스를 따른다. 테스트 프로세스에는 1)공통적으로 수행되는 테스트 작업과 2)개발 방법론, 도메인, 테스트 레벨, 테스트 기법에 따라 다르게 수행되는 테스트 작업이 있다. 본 논문에서는 표준에 명시된 테스트 작업의 공통점과 차이점을 추출하여 테스트 프로세스 아키텍처를 명세하고, 이를 컴포넌트로 구현한 후, 각 컴포넌트가 테스트 작업의 특성에 맞게 조립될 수 있도록 프로세스 수행 도구 개발 방안을 제안한다.

본 논문은 2장의 관련 연구에 이어, 3장에서는 테스트 프로세스 수행 도구 개발 방안을 기술한다. 4장에서는 테스트 프로세스 수행 도구 구현에 대해서 기술하며, 5장에서는 본 도구에 대해 분석한다. 마지막으로 6장에서 결론 및 향후 연구 과제를 제시한다.

2. 관련 연구

본 절에서는 상용 테스트 자동화 도구의 특성에 대해 언급하고, 테스트 프로세스 생성 및 수행 도구에 관한 기존 연구의 특성과 한계점을 명시한다.

(1) 테스트 자동화 도구

프로덕트 라인 개발을 지원하기 위한 아키텍처 추출 및 재공학, 테스팅, 모델링 도구, 프로덕트 라인 스코핑, 형상 관리, 시각화, 링크 도구들이 필요하다[3]. 프로덕트 라인 개발 환경에서는 다차원적인 산출물이 생성되고 재사용 되기 때문에 개발된 산출물이 적합한지 확인하고 검증하는 테스팅이 매우 중요하다[3].

기존의 테스트 도구는 McCabe Test[4]와 같이 테스트 프로세스의 특정 부분만을 지원하는 도구와 Rational Suite TestStudio[5], SilkPlan Pro[6] 등 전체 테스트 프로세스를 지원하는 도구로 분류할 수 있다. Rational Suite TestStudio[5]는 기능성 테스트, 신뢰도 테스트, 성능 테스트를 수행하며 통합 테스트 레벨을 지원한다. SilkPlan Pro[6]는 테스트 프로세스 전체의 모든 테스트 활동을 관리하지만 제공하는 프로세스가 고정되어 있으며 기능 테스트 기법(functional testing)과 회귀 테스트 기법(regression testing), 로드 테스트 기법(load testing)을 제공한다. 테스트 데이타를 생성하고 테스팅을 수행하는 대부분의 도구들은 테스트 프로세스의 특정 부분, 즉 테스트 레벨에 대한 한정된 테스트 기법만을 제공한다. 전체 테스트 프로세스를 지원하는 도구들도 다양한 테스트 레벨에 대한 다양한 테스트 기법을 제공하지 못한다. 또한 도구에서 제공하는 프로세스가 고정적이기 때문에 테일러 된 테스트 프로세스에는 사용하기 어렵다.

본 논문에서 구현하는 Test PET은 개발 방법론과 도메인에 맞는 테스트 프로세스를 쉽게 생성할 뿐만 아니라, 생성된 테스트 프로세스에 따라 테스트를 수행 할 수 있다는 장점이 있다. 또한 특정 테스트 레벨, 테스트 기법만을 사용할 수 있는 것이 아니라 개발 방법론과 도메인에 따른 테스트 레벨 및 기법을 선택할 수 있으며 추가가 용이하다.

(2) 테스트 프로세스 생성 및 수행 도구

소프트웨어 분야가 다양해짐에 따라 다양한 도메인과 어플리케이션 특성에 맞는 개발 프로세스를 정립할 필요가 높아지고 있지만 표준에서 제안하는 개발 프로세스들은 다양한 도메인과 어플리케이션 특성에 따라 적용하기 부족하다. 대부분의 테스트 도구는 고정된 테스트 도구에서 지원하는 프로세스만을 제공하고 프로세스를 생성해 주는 도구들은 프로세스만 생성 할 뿐 생성된 프로세스에 따라 수행하지 못한다. 기존 연구[7]에서는 다양한 표준, 개발 방법론, 도메인에 대한 테스트 프로세스로부터 공통점을 추출하여 테스트 프로세스 메타 모델을 개발하고, 특정 어플리케이션에 맞는 테스트 프로세스를 생성할 수 있는 방안을 제시하였다. 이 방안은 특정 방법론과 도메인에 맞는 테스트 프로세스를 생성하며 이를 재사용 할 수 있지만 생성된 테스트 프로세스에 따라 수행 할 수 있는 환경을 제공하지 못한다. 본 논문에서는 특정 도메인과 어플리케이션 특성에 맞는 테스트 프로세스를 생성하고, 생성된 테스트 프로세스에 따라 수행하는 테스트 프로세스 수행 도구 개발 방안을 제안하고 이를 도구로 개발한다.

3. 테스트 프로세스 수행 도구 개발 방안

본 논문에서는 테스트 계획, 테스트 설계, 테스트 수

행, 테스트 검토로 이루어진 테스트 프로세스를 사용자 요구사항에 맞게 생성하고 생성된 프로세스에 따라 수행하는 테스트 프로세스 수행 도구를 개발한다. 테스트 프로세스 수행도구(Test PET)은 크게 테스트 프로세스 생성 컴포넌트와 테스트 작업 컴포넌트로 이루어진다. 테스트 프로세스 생성 컴포넌트는 도메인과 어플리케이션 특성에 맞는 테일러링 된 테스트 프로세스를 생성하는 컴포넌트이다. 테스트 작업 컴포넌트는 표준에 명시된 테스트 작업을 컴포넌트화 한 것으로 각 테스트 작업에 따라 테스트 계획 생성 컴포넌트, 테스트 설계 생성 컴포넌트, 테스트 수행 컴포넌트, 테스트 검토 컴포넌트가 존재한다.

본 논문에서 제안하는 테스트 프로세스 수행 도구 개발은 그림 1에서처럼 코어 어셋 개발과 제품 개발의 두 단계로 이루어진다.

코어 어셋 개발 단계에서는 테스트 프로세스 수행 도구의 아키텍쳐를 정의하고, 테스트 작업의 공통점과 차이점을 컴포넌트로 명세 및 개발한다. 아키텍쳐는 프로덕트 라인 레벨과 프로덕트 레벨로 세분화하여 개발한다. 프로덕트 라인 레벨에서는 표준에 명시된 테스트 작업의 공통점을 추출하여 테스트 프로세스 생성 컴포넌트를 명세하고, 테스트 작업 컴포넌트를 명세한다. 프로덕트 레벨에서는 테스트 작업의 차이점을 기법과 레벨에 따라 분류하여 컴포넌트로 명세 한다. 컴포넌트 개발 단계에서는 명세 한 컴포넌트를 구현하고 생산 계획 개발 단계에서는 컴포넌트를 조립할 방안을 세운다.

개발 제품 개발 단계에서는 코어 어셋 개발 단계에서 구현한 컴포넌트를 생산 계획에 따라 조립한다.

3.1 코어 어셋 개발 단계

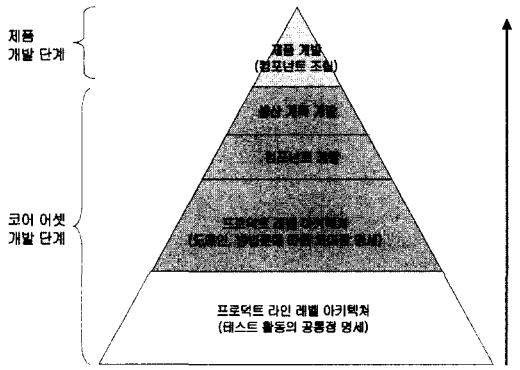


그림 1 테스트 프로세스 수행 도구 개발 단계

코어 어셋은 프로덕트 라인에서 구조적인 재사용을 위해 구현된 소프트웨어 컴포넌트 뿐 아니라 프로덕트 라인에서 제품이 공유해야 할 아키텍쳐를 포함한다[1]. 프로덕트 라인 아키텍쳐는 일반적으로는 프로덕트 라인의 필요성을 만족시키면서, 프로덕트 라인 범위 안에서 제품의 차이점을 제공하기 위해 필요한 차이점들을 명시적으로 허용할 수 있어야 한다[1,8]. 또한 제품의 구조를 명시하고 어셋의 기본이 되는 컴포넌트를 위한 인터페이스 명세를 제공한다.

코어 어셋 개발 단계는 '프로덕트 라인 레벨 아키텍쳐 개발', '프로덕트 레벨 아키텍쳐 개발', '컴포넌트 개발', '생산 계획 개발'로 이루어진다.

3.1.1 프로덕트 라인 레벨 아키텍쳐 개발 단계

프로덕트 라인 레벨 아키텍쳐는 테스트 작업의 공통점으로 이루어진다. 프로덕트 라인 레벨 아키텍쳐 개발은 테스트 프로세스 생성을 위한 '테스트 프로세스 생성 컴포넌트 명세'와 테스트 작업의 공통점을 컴포넌트로 명세하는 '테스트 작업 컴포넌트 명세' 단계를 갖는다.

• 테스트 프로세스 생성 컴포넌트 명세

본 연구진은 테스트 프로세스를 메타 모델로 정의하여 개발 방법론, 도메인 및 특정 어플리케이션에 맞는 테스트 프로세스를 생성할 수 있는 방안을 제안하고, '테스트 프로세스 생성 도구'를 개발[7]하였다. 본 논문에서는 테스트 프로세스를 생성하는 것에서 더 나아가 이를 수행할 수 있도록 한다. 테스트 프로세스 생성 컴포넌트 명세 단계에서는 [7]의 방안에 따라 테스트 프로세스 생성 컴포넌트를 명세한다. 테스트 프로세스 생성 컴포넌트는 XML[9]로 변환된 테스트 프로세스 메타 모델에 방법론과 도메인 플러그 인을 맞춤하여 테일러링 된 프로세스를 생성한다. 그림 2와 같이 테스트 프로세스 생성 컴포넌트는 맞춤을 수행하는 'Process-

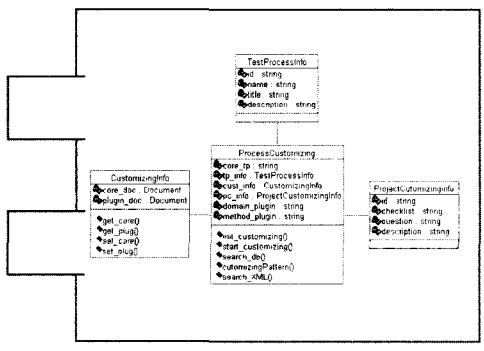


그림 2 테스트 프로세스 생성 컴포넌트 명세

Customizing' 클래스, 프로세스 정보를 담는 'TestProcessInfo' 클래스, 테일러링을 정보를 담는 'CustomizingInfo' 클래스, 어플리케이션 정보를 나타내는 'ProjectCustomizingInfo' 클래스로 이루어진다.

• 테스트 작업 컴포넌트 명세

테스트 표준[10,11]으로부터 개발 방법론이나 도메인에 공통적으로 필요한 테스트 계획, 테스트 설계 생성, 테스트 수행, 테스트 검토에 관한 핵심 테스트 작업을 추출한다. 테스트 작업은 산출물을 작성하는 부분과 테스트 절차를 수행하는 부분으로 나눌 수 있다. 따라서 테스트 작업 컴포넌트는 그림 3에서처럼 산출물 작성 컴포넌트와 테스트 절차 수행 컴포넌트로 구성된다. 그림 3은 테스트 작업 중 테스트 설계 생성 부분에 대해 컴포넌트로 명세한 것이다.

프로덕트 라인 레벨 아키텍처는 테스트 프로세스 생성 컴포넌트와 테스트 계획, 테스트 설계 생성, 테스트 수행, 테스트 검토에 대한 테스트 작업 컴포넌트로 이루어진다. 프로덕트 라인 아키텍처에 개발 방법론과 도메인에 따른 차이점을 반영하는 것은 테스트 절차 수행 컴포넌트를 통하여 '프로덕트 레벨 아키텍처 개발 단계'에서 이루어진다.

3.1.2 프로덕트 레벨 아키텍처 개발 단계

프로덕트 레벨 아키텍처 개발 단계에서는 특정 개발 방법론과 도메인을 수용할 수 있도록 프로덕트 라인 레벨 아키텍처의 테스트 절차 수행 컴포넌트를 구체화한다. 방법론과 도메인에 따른 차이점은 표 1과 같다. 테스트 레벨에 따라 테스트 수행 작업이 달라지며, 테스트 기법에 따라 테스트 설계 생성 작업이 달라진다. 차이점으로 인해 변경되는 테스트 절차 수행 컴포넌트들을 프로덕트 라인 레벨 아키텍처의 컴포넌트를 상속한 후 이를 차이점들의 특성에 맞게 구체화하여 프로덕트 레벨의 아키텍처를 생성한다.

표 1 차이점과 테스트 작업

[illegible]	[illegible]	정보내용
테스트 레벨	테스트 수행	테스트 절차 수행 컴포넌트
테스트 기법	테스트 설계 생성	테스트 절차 수행 컴포넌트

테스트 설계 생성의 테스트 절차 수행 컴포넌트에는 테스트 기법 컴포넌트들을, 테스트 수행의 테스트 절차 수행 컴포넌트에는 테스트 환경 설정 컴포넌트들을 연결한다. 그림 4는 그림 3의 테스트 절차 수행 컴포넌트에 컴포넌트 관련 테스트 기법인 'Component Qualification Technique', 'Component Customization Technique', 'Component Composition Technique', 'Component Qualification Technique'을 추가한 프로덕트 레벨 아키텍처이다.

3.1.3 컴포넌트 개발 단계

컴포넌트 개발 단계에서는 프로덕트 레벨 아키텍처의 컴포넌트들을 구현한다. 본 논문에서는 엔터프라이즈 자바 빈즈(Enterprise Java Beans)를 사용하여 컴포넌트

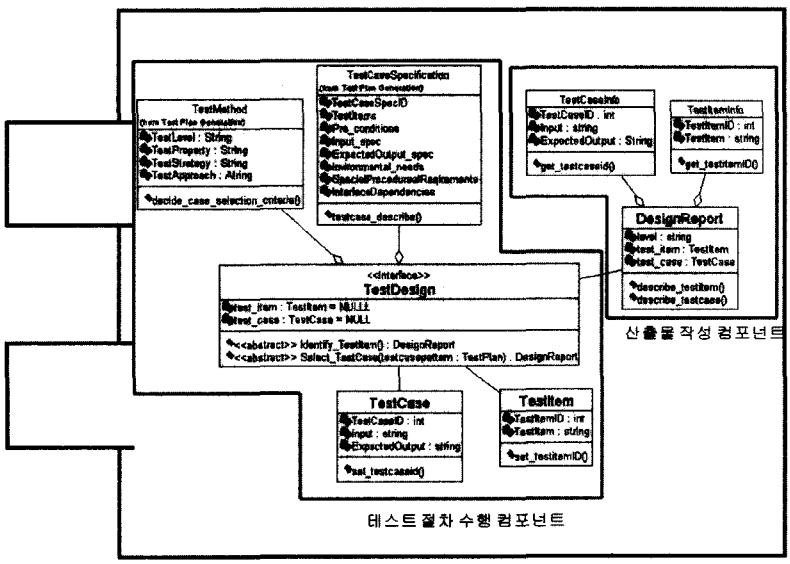


그림 3 테스트 설계 생성 컴포넌트 명세

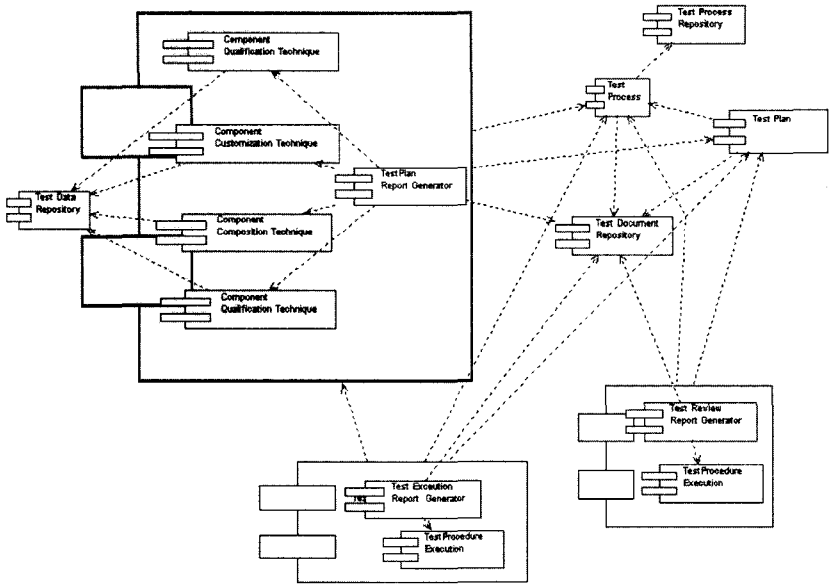


그림 4 프로덕트 레벨 아키텍처 - 컴포넌트 다이어그램

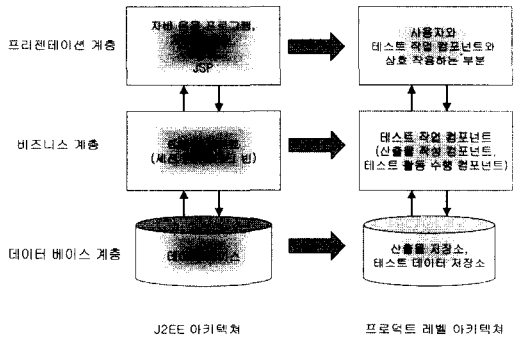


그림 5 J2EE 아키텍처와 프로덕트 레벨 아키텍처 매핑

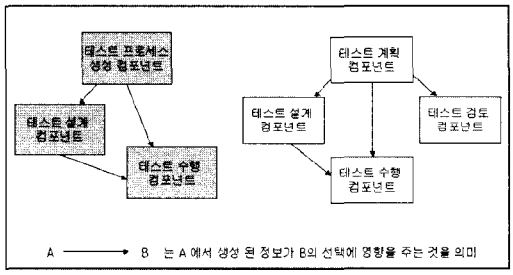


그림 6 테스트 작업 컴포넌트 연관 관계

를 구현하였다. J2EE 아키텍처에 프로덕트 레벨 아키텍처를 맵핑하면 그림 5에서처럼 프리젠테이션 계층은 사용자 인터페이스로, 비즈니스 계층은 테스트 작업 컴포넌트로, 데이타 베이스 계층은 산출물 저장소와 테스트 데이타 저장소로 맵핑된다. 테스트 작업 컴포넌트의 산출물 작성 컴포넌트는 세션 빈으로, 테스트 절차 수행 컴포넌트는 세션 빈으로, 산출물이나 테스트 데이타 등의 결과물들은 엔티티 빈으로 표현한다.

3.1.4 생산 계획 개발 단계

생산 계획 개발 단계에서는 컴포넌트 개발 단계에서 구현한 테스트 작업 컴포넌트들을 조립한다. 테스트 작업 컴포넌트 조립을 위해서는 컴포넌트의 정보를 분류 및 저장한 후 테스트 작업 컴포넌트를 연결하는 연결자를 생성해야 한다. 프로덕트 레벨 아키텍처에서 개발 방법론과 도메인에 따라 추가된 테스트 기법 컴포넌트들과 테스트 환경 설정 컴포넌트들의 정보를 방법론과 도메인 각각에 대해 테스트 레벨, 테스트 기법으로 분류하여 데이타 베이스에 저장한다. 그림 6은 각 테스트 작업 컴포넌트 사이의 연관 관계를 보여준다. 테스트 작업 컴포넌트 연결자는 서로 연관 관계가 있는 테스트 작업 컴포넌트를 연결해 주고 테스트 작업 컴포넌트가 수행될 수 있게 하는 역할을 한다.

3.2 제품 개발 단계

제품 개발 단계에서는 코어 어셋 개발 단계에서 개발된 컴포넌트를 생산 계획에 따라 맞춤 및 조립하여 테스트 프로세스 수행 도구를 개발한다.

4. Test PET - 테스트 프로세스 수행 도구 구현

본 논문에서는 3장에서 제안한 테스트 프로세스 수행 도구 개발 방안을 기반으로 Test PET 도구를 구현하였

다. Test PET는 사용자(테스터)로부터 개발 방법론, 도메인 및 어플리케이션 정보를 입력받아 이에 맞는 테스트 프로세스를 생성하며, 생성된 테스트 프로세스에 따라 테스트 산출물을 작성할 수 있고, 저장 및 관리할 수 있으며, 방법론과 도메인에 적합한 테스트 기법을 선택하여 실행할 수 있도록 한다. 즉, Test PET은 사용자의 요구에 맞는 테스트 프로세스를 생성하고 실행한다.

Test PET의 구성은 그림 7과 같으며, Window 2000 Professional 환경에서 Java 2 Platform, Enterprise Edition version 1.3을 기반으로 Java 2 Platform, Standard Edition version 1.3.1로 구현하였다. 사용된 데이타 베이스는 Oracle 8i Enterprise Edition Release 3 (8.1.7) for Windows 2000/NT이다.

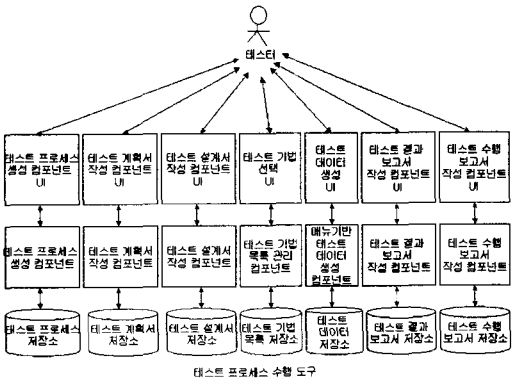


그림 7 Test PET 구성 컴포넌트

4.1 Test PET의 실행 사례

본 절에서는 사용자(테스터)의 입장에서 Test PET을 이용하여 테스트 프로세스를 생성하고, 수행하는 사례를 기술한다. 사용자의 요구사항은 다음과 같다고 가정한다.

개발 방법론 : Rational Objectory

도메인 : E-Commerce

테스트 레벨 : 시스템 테스트 레벨

테스트 기법 : 품질 인증 테스트 기법[12]

사용자가 요구사항에 따라 Test PET을 실행하는 순서는 그림 8과 같이 1) 테스트 프로세스 생성, 2) 테스트 계획서 작성, 3) 테스트 기법 선택, 4) 테스트 데이타를 생성, 5) 테스트 설계서 작성, 6) 테스트 결과 보고서 작성, 7) 테스트 수행 보고서 작성이다.

4.1.1 테스트 프로세스 생성

사용자가 'Process Generation' 메뉴를 선택하면 방법론과 도메인 및 특정 어플리케이션 정보를 입력하는 화면이 나타난다(그림 9의 ①). Objectory 방법론과 E-Commerce 도메인을 선택하고 어플리케이션 정보를 입력한다. Test PET은 입력된 내용을 바탕으로 XML 형식의 테스트 프로세스를 생성하여 보여준다. 'View Process List' 메뉴를 통하여 생성된 프로세스의 목록을

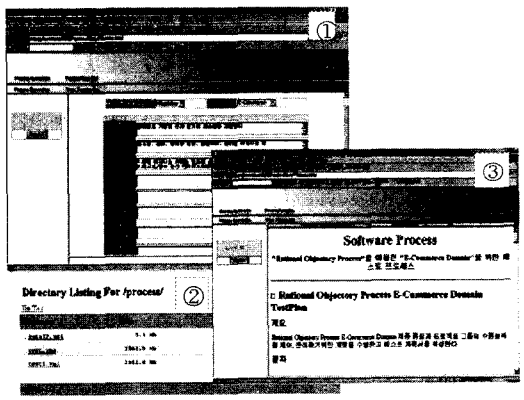


그림 9 테스트 프로세스 생성

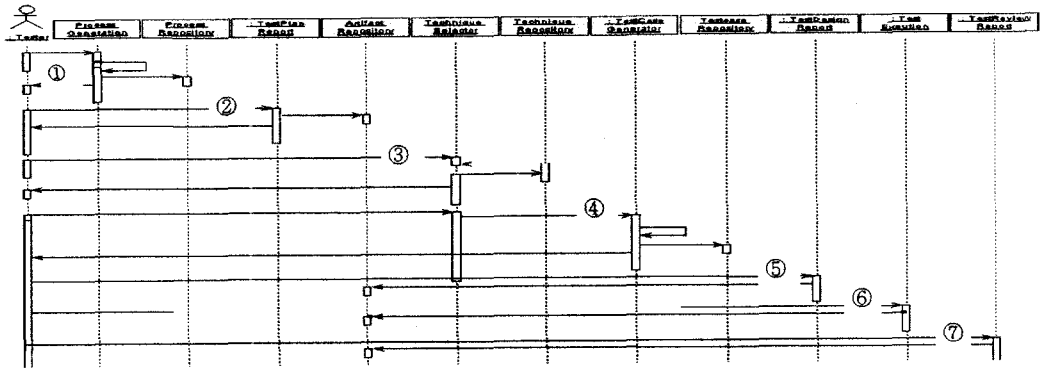


그림 8 Test PET 실행 순서

볼 수 있다(그림 9의 ②). 사용자는 생성된 테스트 프로세스를 하위 프레임에서 볼 수 있으며 테스트 수행 시 가이드로 참조할 수 있다(그림 9의 ③).

4.1.2 테스트 계획서 작성

사용자는 'Test Plan Generation' 메뉴를 통하여 테스트 계획서를 작성하고 관리할 수 있다. 'Process Execution' 메뉴에서 'Test Plan Generation' 메뉴를 선택하면 테스트 계획서의 목록이 나타난다(그림 10의 ①). Write를 선택하여 테스트 계획서를 작성한다(그림 10의 ②). 제공되는 문서의 형식은 표준[12]에 맞추어져 있다. 작성된 테스트 데이타는 그림 10의 ③과 같다.

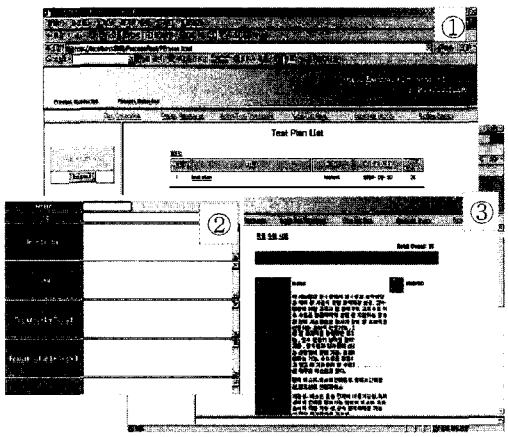


그림 10 테스트 계획서 작성

4.1.3 테스트 기법 선택

사용자는 'Select Test Technique' 메뉴를 선택하여 개발 방법론과 개발 도메인에 해당하는 테스트 기법을 선택한다. 그림 11은 사용자가 입력한 요구사항 즉, E-Commerce 도메인과 Objectory 방법론에 대한 테스트 기법의 목록을 보여준다. ①에서 ec_plugin_1(E-Commerce 도메인)을 선택하면 목록이 나타난다. 테스트 기법의 목록(②)은 프로세스에 나타난 테스트 기법(③)과 동일하다.

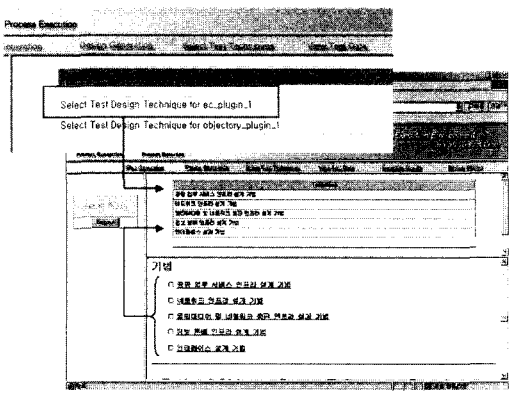


그림 11 테스트 기법 선택

4.1.4 테스트 데이타 생성

사용자는 테스트 레벨과 테스트 기법을 선택한다. 그림 12의 ①에서처럼 시스템 레벨을 선택하면, 시스템 레벨에 속한 테스트 기법의 목록이 나타난다. 예를 들어 '품질 인증을 위한 메뉴 기반 테스트 기법(Menu-based Testing)'을 선택한 후 Execute 버튼을 누르면 선택한 테스트 기법에 해당하는 테스트 도구가 실행되어 ②와 같은 인터페이스가 나타난다. 사용자는 인터페이스를 통하여 테스트 데이타를 생성한다. 생성된 테스트 데이타는 'View Test Data' 메뉴를 통하여 볼 수 있다(그림 12의 ③).

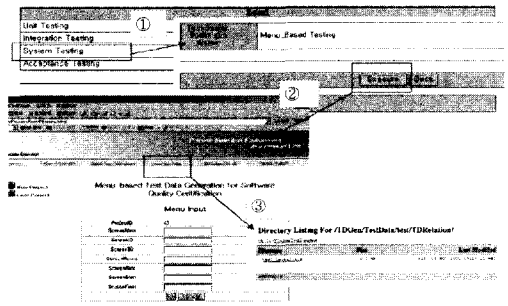


그림 12 테스트 기법 선택, 수행 및 테스트 데이타 보기

(5) 테스트 설계서, (6) 테스트 결과 보고서, (7) 테스트 수행 보고서 작성은 (1)의 테스트 계획서 작성과 동일하다.

5. 분 석

Test PET은 전체 테스트 프로세스를 관리 및 수행하는 도구이다. 따라서 전체 테스트 프로세스를 관리하는 도구인 Rational Suite TestStudio[5], SilkPlan Pro[6]와 비교 분석한다. 프로덕트 라인 지원 도구가 갖춰야 할 특성[3]인 1) 산출물의 상호 운영성, 2) 추적 가능성, 3) 표준과 일치, 4) 테일러 가능한 프로세스, 5) 확장가능성 측면에서 분석한다. 분석결과는 표 2와 같다.

5.1 산출물의 상호 운용성

다른 도구에서 생성된 산출물을 처리할 수 있어야 하고 다른 도구에 입력이 가능한 산출물을 생성할 수 있어야 한다. Test PET은 산출물을 테이블을 이용하여 데이타 베이스에 저장한다. 테이블로 명세된 산출물은

표 2 Test PET과 타 도구들과의 비교 분석

특징	Rational Suite TestStudio	SilkPlan Pro	Test PET
산출물의 상호 운용성	용이하지 않음	용이하지 않음	용이함
추적 가능성	용이	용이	용이
표준과 일치	일부 일치	일부 일치	일치
테일러 가능한 프로세스	지원하지 않음	지원하지 않음	지원
확장 가능성	용이하지 않음	용이하지 않음	용이 (다양한 테스트 기법 등 추가 가능)

쉽게 XML로 변경 가능하다. XML 문서는 플랫폼 독립적이며, 다양한 형식을 가질 수 있으며, 데이타를 검색이 용이하다는 장점이 있기 때문에 산출물을 관리하는데 용이한 형식이다. Rational Suite TestStudio와 SilkPlan Pro는 도구에서 제공하는 파일의 형식으로 산출물을 생성한다.

5.2 추적 가능성

산출물의 변경 시 쉽고 빠르게 연관된 다른 산출물을 변경할 수 있어야 한다. Test PET은 산출물과 관련된 다른 산출물을 쉽게 찾을 수 있도록 한다. Rational Suite TestStudio와 SilkPlan Pro도 모두 프로젝트 단위로 관리하기 때문에 산출물에 대한 정보를 쉽게 찾을 수 있다.

5.3 표준과 일치

모든 도구는 표준에 일치하고 지원할 수 있어야 한다. Test PET은 표준을 근거로 테스트 작업의 공통점과 차이점을 분석하여 구현한 도구이다. Rational Suite TestStudio와 SilkPlan Pro는 테스트 작업의 핵심적인 항목만을 제공하기 때문에 일부만 표준에 상응한다.

5.4 테일러 가능한 프로세스

개발 프로세스에 맞게 도구에서 제공하는 프로세스가 테일러 가능해야 한다. Test PET은 테일러 가능한 테스트 프로세스를 생성하며, 생성된 테스트 프로세스에 따라 수행할 수 있는 환경을 제공한다. Rational Suite TestStudio은 Rational Unified Process[15]를 도구에 포함하여 지원하고 있으나 변경은 가능하지 않다. SilkPlan Pro도 개발 프로세스를 제공하지만 고정되어 있다.

5.5 확장 가능성

테일러 가능한 프로세스를 지원하려면 도구가 확장 가능해야 한다. Test PET은 컴포넌트로 개발되었기 때문에 다른 컴포넌트와 조립함으로써 확장이 가능하다. Rational Suite TestStudio와 SilkPlan Pro은 기능의 변경이나 추가가 용이하지 않다.

6. 결론 및 향후 연구 과제

테스트 프로세스를 지원하기 위해 상용화 된 다양한 테스트 도구들이 존재하지만 개발 환경에 맞도록 테일러링 된 테스트 프로세스를 생성하고 이를 수행 할 수 있는 테스트 도구는 개발되지 않았다. 본 논문에서는 테스트 프로세스 수행 도구 개발 방안을 제안하였고 제안한 방안으로 Test PET을 개발하였다. Test PET은 개발 방법론, 도메인 및 특정 어플리케이션에 적합한 테스트 프로세스를 생성하고 생성한 프로세스에 따라 테스트를 수행 할 수 있는 환경을 제공한다. 즉, Test PET은 생성한 테스트 프로세스에 따라 테스트 계획부터 테스트 결과보고서 작성까지의 모든 테스트 작업을 수행하고 산출물을 생성하며, 개발 방법론과 도메인에 맞는 테스트 기법에 의한 테스트 데이타 생성까지 지원하는 테스트 도구이다.

본 논문에서 구현한 Test PET은 3장에서 제안한 테스트 프로세스 수행 도구 개발 방안이 실제 개발에 적용 가능함을 보여주기 위하여 특정 개발 방법론에 해당하는 일부 테스트 기법만을 지원할 수 있도록 개발되었다. 좀 더 다양한 개발 방법론, 도메인에 해당하는 테스트 기법들을 지원할 수 있도록 코어 어셋을 추가 개발하여야한다. 또한 현재 일부 수동적으로 이루어지는 테스트 프로세스 수행을 자동화할 수 있도록 보완하는 것이 필요하다.

참 고 문 헌

[1] Paul Clements, Linda Northrop, "A Framework for Software Product Line Practice, Version 3.0," SEI/CMU, March 2002.

[2] John D. McGregor, "Testing a Software Product Line," SEI Technical report CMU/SEI-2001-TR-022, December 2001.

[3] Len Bass, Paul Clements, Patrick Donohoe, John D. McGregor, Linda Northrop, "Fourth Product Line Practice Workshop Report," SEI Technical report CMU/SEI-2000-TR-002, February 2000.

[4] http://www.mccabe.com/mccabe_test.php

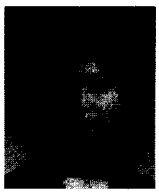
[5] http://www.rational.co.kr/product/prd_testStudio.asp?phaseCode=Suite

[6] http://www.segue.com/html/s_solutions/s_silkplan/index.htm

[7] Jooyoung Seo, Yoonjung Lee, Byoungju Choi, "A Scheme on Software Process Component Reuse for Product Line Practice," International Journal of Computer & Information Science, Vol.3 No.1, pp. 41-48, 2002.3.

[8] Paul Clements, Linda Northrop, "Software Product Lines - Practices and Patterns," Addison-Wesley, August 2001.

[9] W3C, Extensible Markup Language (XML) 1.1, http://www.w3c.org/XML, 1998.
[10] ISO/IEC 12207 : Information Technology - Software Life Cycle Process.
[11] ANSI/IEEE Std 1012-1986 IEEE Standard for Software Verification and Validation Plan.
[12] Yoonjung Lee, Eunjung Chun, Byoungju Choi, "A Menu-based Test Generation Technique for Software Quality Certification," the 8th ISSAT Int. Conference on Reliability and Quality in Design (ISSAT 2002: California USA), pp191-195, Aug. 2002.
[13] ANSI/IEEE Std 829-1983 IEEE Standard for Software Test Documentation.
[14] http://www.segue.com/html/s_solutions/s_silkplan/index.htm
[15] http://www.rational.com/products/rup/index.jsp



천 은 정

1995년~2001년 이화여대 독어독문 학사 (컴퓨터학과 부전공). 2001년~2003년 이화여대 컴퓨터학과 석사. 2003년~현재 삼성전자 CTO 전략실 소프트웨어 센터 관심분야는 Software Testing, Product-line Engineering. Software Quality Assurance, Embedded Software

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 교수. 관심분야는 소프트웨어공학, 소프트웨어 테스트, 소프트웨어 및 데이타 품질 측정