

Scenario Framework for ERP Testing and Training: SF-ETT

박광호^a, 김호택^a, 안윤지^a

^a한양대학교 일반대학원 e-business 경영학과

426-791, 경기도 안산시 사1동 1271

Tel: +82-31-400-5644, Fax: +82-31-400-5591, E-mail: oobepark@hanyang.ac.kr

Tel: +82-11-792-8791, Fax: +82-31-400-5591, E-mail: mangwul@hanyang.ac.kr

Tel: +82-10-3299-4917, Fax: +82-31-400-5591, E-mail: yunjiahn@hanyang.ac.kr

Abstract

Effective Training has been recognized as one of the most important success factors for enterprise resource planning (ERP) system operations. However, both ERP system vendors and user companies have failed to provide an effective training method for users because practical business process cases cannot be formalized. Also, incomplete testing by ERP system vendors results in insufficient and ineffective user training.

This paper suggests a scenario framework for ERP testing and training (SF-ETT). SF-ETT is constructed by expanding the practice-based ERP testing scenario construction framework designed for uniERP/II. SF-ETT contains concrete business process entities that users understand actually and provides practice definition, notation, and architecture.

Keywords:

ERP, Testing, Training, Framework

1. 서론

ERP(Enterprise Resource Planning) 시스템은 본질적으로 최상의 비즈니스 프로세스로 설계되었으므로 이를 도입하는 기업은 전략, 조직 구조 및 문화를 변화시킬 수 있는 영향력을 가지고 있다 (Devenport, 1998). 성공적인 ERP 시스템 운영에 있어 사용자 교육의 중요성은 여러 선행 연구에서 밝혀진 바 있다 (윤종수 등, 1998; Bingi et al., 1999; Gupta, 2000, 이석준, 2001; 황재훈과 이선로, 2002; 박광호 등, 2006) 특히, McAlary (1999)는 ERP 시스템 운영의 실패요인으로 불충분한 교육훈련으로 제시하고 있다. Kapp (2001)은 Dow Chemical, Dell Computer, Hershey Foods 등 많은 기업들이 사용자에게 대한 교육 계획이나 주제를 제대로 맞추지 못하여 안정화에 난항을 겪었다는 사실을 언급하며 ERP 교육의 중요성을 강조한 바 있다.

ERP 시스템의 성공적인 도입 및 운영에 있어 교육의 역할이 매우 중요하다. 이는 ERP 시스템이 영업, 생산, 구매, 재고, 회계 등 회사 내의 비즈니스

프로세스를 통합하고, 모든 거래 데이터는 통합 처리되기 때문이다.

ERP 사용자는 자신이 ERP 시스템에서 실행해야 할 거래 처리와 연관된 프로세스에 대해 가장 이해를 수월하게 할 수 있다. 그러나, ERP 시스템 벤더는 고객의 실제 거래 처리 과정에 대해 구체적으로 교육하거나 이에 필요한 정보를 제공하지 못하고 있는 실정이다 (황재훈, 2002). 프로세스 실제 처리 사례는 기업마다 다르기 때문에 이를 ERP 시스템 벤더가 표준화, 구체화 하기에 어려움이 있기 때문이다. 또한, 도입 기업도 실제 프로세스 처리 과정에 대해 전사적으로 정보를 공유하지 못하고 있다. 결국 ERP 시스템 벤더와 도입 기업 모두 실제 프로세스 처리 사례의 정형화된 정보를 제공하지 못해 사용자에게 심도 있는 프로세스 교육 방안을 제시 못하고 있는 실정이다.

일반적으로 ERP 시스템에 대한 테스트는 사용자 인터페이스를 대상으로 실행된다. 사용자 인터페이스는 분석 단계에서 정의된 유스 케이스 (Use Case) (Jacobson et al., 1992)를 가시적 구현한 것이다. 완벽한 사용자 인터페이스에 대한 테스트는 실질적으로 사용자가 거래 처리할 수 있는 모든 사례를 대상으로 실시되어야 한다. 만일, 이와 같이 모든 사례에 대한 테스트가 적절히 실행되지 않았다면 결국에는 사용자가 거래 처리하는 과정에서 테스트를 하는 결과를 초래할 것이다.

효과적인 테스트를 위해서는 유스 케이스에 대한 시나리오를 상태 중심, 데이터 중심으로 구체적이며 결과 지향적으로 작성해야 한다 (박광호, 1999).

ERP 교육에 있어도 시나리오를 기반으로 한 시스템이 효과적이라는 연구가 발표된 바 있다 (김훈태 외, 1999). 기업의 가상상황을 시나리오로 구성하고 CBT(Computer-Based-Training) 기반으로 구축하여 시나리오에 따라 기업의 표준적인 비즈니스 프로세스와 ERP의 사용자 화면을 연동시킴으로써 ERP 사용자가 기업의 비즈니스 프로세스를 이해하고, 비즈니스 프로세스에 따른 ERP의 사용법과 작용 원리를 습득 할 수 있도록 지원하는 것이 효과적인 ERP 교육 방법이라는

주장이다.

ERP 시스템에 대한 벤더의 불완전한 테스트는 결국 불충분하고 비효율적인 사용자 교육으로 이어진다. 발생 가능한 모든 사례에 대한 완벽한 준비가 없기에 실제 교육은 일반적인 거래 처리 사례를 중심으로 진행되기 때문이다. 따라서, ERP 도입 및 운영의 성공을 위해서는 사전에 충분한 테스트와 이를 기반으로 한 교육이 수반되어야 한다..

본 논문에서는 ERP 시스템의 테스트와 교육용 시나리오를 작성하기 위한 프레임워크인 SF-ETT (Scenario Framework for ERP Testing and Training)를 제시하고 있다. SF-ETT는 ERP시스템의 사용자가 실질적으로 이해하는 비즈니스 프로세스의 실체를 기반으로 구성되어 프랙티스 (Practice) 정의, 표기 방법, 프레임워크 구조를 제시한다. SF-ETT는 새로운 ERP 시스템 교육 방향 제시하여 비즈니스 프로세스의 대한 이론적 교육과 실무적 교육의 병행 효과를 얻을 수 있다..

SF-ETT는 삼성SDS의 ERP 솔루션인 uniERPII를 위한 통합 테스트 시나리오 프로젝트를 통해 정립된 프랙티스 기반 ERP 테스트 시나리오 작성 프레임워크를 확장한 것이다. SF-ETT는 uniERPII 시스템의 지속적인 테스트와 사용자 교육을 위한 uTF (uniERP Testing & training Framework)로 적용 중이다. 본 논문에서 제시하는 예들은 uTF에 설계 내역을 중심으로 설명한 것임을 밝힌다.

2. 프랙티스: ERP 테스트 및 교육 시나리오 작성을 위한 기반 구조

2.1 프랙티스 정의의 필요성

ERP 시스템은 기업의 비즈니스 프로세스를 통합 처리하는 시스템이다. 비즈니스 프로세스란 조직의 목표 달성을 위하여 다양한 비즈니스 규칙에 의해 정의된 상호연관이 있는 비즈니스 기능의 집합을 뜻하며 (김민수, 2004), 비즈니스 전반에 관한 특성, 수행되는 활동에 대한 특성, 진행조건을 포함하고 있다 (채정숙과 박종홍, 2005). ERP 시스템은 이와 같은 비즈니스 프로세스를 프로그램의 집합으로 구성해 놓은 것이라 할 수 있다.

그러나, ERP 시스템을 도입하는 기업은 ERP의 표준 비즈니스 프로세스를 자사 특성에 맞게 적용하게 된다. 따라서, 표준 비즈니스 프로세스 보다는 적용된 (Customized) 비즈니스 프로세스를 중심으로 한 시나리오가 테스트이나 교육에 더욱 효과적이다 (박광호와 서형교, 2006). 효과적인 테스트이나 교육 시나리오는 ERP의 표준 프로세스가 아니라 사용자가 실제로 처리해야 할 비즈니스 프로세스에 대한 절차와 방법에 대한 것이다. 우리는 ERP의 도입된 비즈니스 프로세스

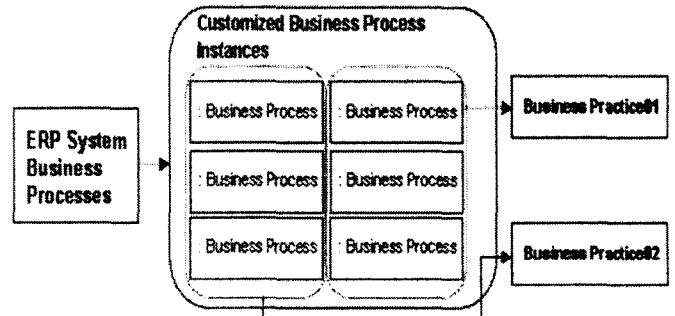
사례를 프랙티스라고 정의하고 이를 기반으로 한 테스트 및 교육 시나리오 프레임워크를 제시하고자 한다.

2.2 프랙티스 정의

일반적으로 ERP를 도입하는 기업은 초기에 커스터마이징 작업을 하게 되는데 이 작업은 ERP 표준 프로세스에 기준정보, 형상정보, 비즈니스 규칙을 설정하여 맞춤형 프로세스로 전환시키는 것으로 볼 수 있다. 이와 같은 맞춤형 프로세스로 ERP가 도입된 후에는 [그림 1]과 같이 실제로 업무 담당자가 처리하는 사례의 유형별로 각각 독립적으로 정의될 수 있는데 우리는 이와 같은 프로세스별 사례 유형을 프랙티스라고 정의하기로 한다.

프랙티스는 초기 시스템 설정 (Setup) 정보를 기반으로 실행된다. 시스템 설정 정보는 (1) 품목, 거래처, BOM (Bill Of Materials) 등 기준 정보, (2) 여신한도승인, 검사여부 등의 비즈니스 규칙, (3) 수주형태, 수불형태, 전표 분개 형태 등 형상 (Configuration) 정보, (4) 거래 처리 담당자 및 부서 등 조직 정보 등으로 분류된다. 또한, 프랙티스의 수행 결과, 거래 데이터가 발생하게 되는데, 선, 후행 거래 데이터와 연관성을 가지게 된다. 또한, 회계적 거래인 경우에는 회계전표가 자동으로 등록되기도 하고 한 기능 영역에서 발생한 거래가 타 기능 영역에 관리 목적상 중복해서 거래가 등록되는 경우도 있다. 따라서, 프랙티스 수행 결과로 발생하는 거래 데이터와 이들 상호간의 연관 관계를 명확하게 설명할 필요가 있다.

이와 같이 프랙티스는 도입 기업의 시스템 설정, 이를 기반으로 한 유관 거래의 순차적 실행, 실행 결과 발생하는 거래 데이터와 상호 연관관계를 체계적으로 추상한 것이라고 정의할 수 있다.



[그림 1] 비즈니스 프로세스와 프랙티스

2.3 계층적 프랙티스 구조

프랙티스에 대한 테스트와 교육 시나리오는 단위 프랙티스에서 통합 프랙티스까지 4 레벨로 작성된다. 단위 프랙티스 (Unit Practice) 시나리오는

일반적으로 단위 프로그램의 개별적 기능을 테스트하고 교육하기 위한 목적으로 작성된다. 단위 프랙티스 시나리오는 개별 프로그램의 기능이 정확히 구현되었는가를 검증하고 이를 교육하는 것으로 가장 단순한 수준인 레벨 0 프랙티스 (Level 0 Practice : L0)로 정의된다. 수주등록, 출하등록, 매출채권등록 등과 같은 단위 프로그램에 대한 시나리오가 이에 속한다.

레벨 1 통합 프랙티스 (Level One Integrated Practice: L1) 시나리오는 단위 거래의 완결을 목적으로 수행되는 2~3개 L0 프랙티스의 조합으로 작성된다. L1 프랙티스의 대표적인 예는 전표처리 (전표등록=>전표내역등록=>전표승인), 수주처리 (수주등록=>수주내역등록=>수주승인), 출하처리 (출하요청=>출하내역등록=>출고등록) 등과 같이 단위 거래를 완전하게 처리하는 프랙티스를 들 수 있다. L1 프랙티스는 이와 같은 단위 거래의 완전한 처리에 처리 결과를 확인하는 조회나 출력 프로그램까지 포함 한 형태로 정의될 수도 있다. 예를 들어, 수주처리 프랙티스는 수주 거래 처리 결과 까지 확인하기 위한 조회, 출력 프로그램까지 포함하여 수주처리 (수주등록=>수주내역등록=>수주승인=>수주현황조회=>수주대장출력=>재고현황조회) 등으로 정의될 수도 있다.

레벨 2 통합 프랙티스 (Level Two Integrated Practice: L2) 시나리오는 단위 기능 영역내의 프랙티스들의 집합적 거래 처리에 대한 절차와 방법에 대한 것이다. 영업관리 (수주처리=>출고처리=>매출처리=>계산서처리=>입금처리), 생산관리 (생산계획=>지시계획=>작업지시처리=>부품출고처리=>공정실적처리=>공정검사처리=>생산입고처리) 등은 단위 기능 영역내의 거래의 집합적 완결 처리를 목적으로 하는 시나리오이다.

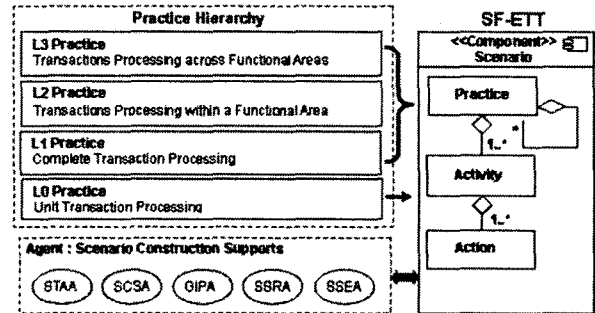
레벨 3 통합 프랙티스 (Level Three Integrated Practice: L3) 시나리오는 다수의 기능 영역간의 관계 속에서 집합적으로 거래를 완결 처리하는 수준에서 작성된다. 예를 들어, 긴급수주에 의한 작업지시 및 판매 프랙티스는 (기초재고처리=>영업관리=>생산관리=>회계관리) 등 총 4개 기능 영역간의 L2 프랙티스 시나리오를 연결시켜 긴급수주로 유발된 모든 거래를 완결 처리하는 시나리오이다.

3. 시나리오 프레임웍

3.1 구조

SF-ETT는 [그림 2]와 같이 시나리오 (Scenario) 컴포넌트와 시나리오 작성 지원을 위한 에이전트 (Agent)로 구성되어 있다. 우선, 시나리오

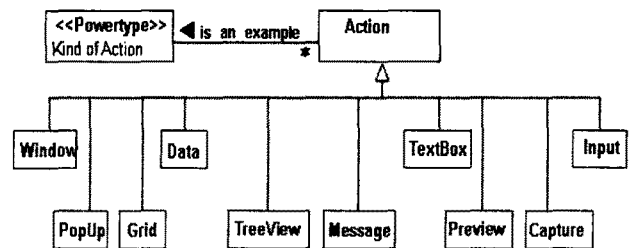
컴포넌트는 비즈니스 프로세스의 정형화된 구체적 실행 사례인 프랙티스를 대상으로 설계되었다. 즉, SF-ETT는 L0, L1, L2, L3 등 4계층 프랙티스에 대한 시나리오를 각각 프랙티스 (Practice)와 액티비티 (Activity), 그리고 액션 (Action) 클래스로 구성하도록 설계되었다.



[그림 2] SF-ETT 구조

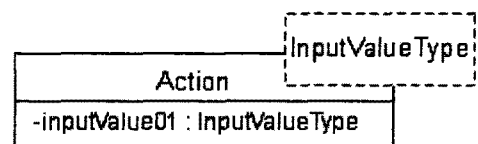
3.2 액션 (Action)

액션은 ERP 시스템 운영에 있어 가장 기본적인 단위 작업이다. 액션은 GUI (Graphical User Interface) 작업과 그 작업에 필요한 입력 데이터를 기본 속성으로 정의할 수 있다. 액션은 [그림 3]에서 분류된 것처럼 사용자가 키보드와 마우스로 수행하는 단위 작업인 것이다.



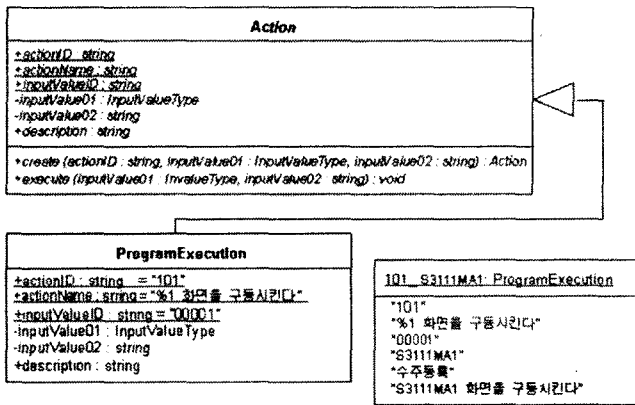
[그림 3] 액션 타입

GUI 작업은 구체적으로 지정하기 위해 기본적으로 2가지 입력 데이터, inputValue01과 inputValue02를 사용하게 된다. 단순 탭 이동인 경우에는 integer 타입이 사용되겠지만 데이터를 입력의 경우에는 integer, date, string 등 다양한 데이터 타입이 사용된다. 따라서, 액션은 [그림 4]와 같이 inputValue01의 데이터 타입을 수용할 수 있는 파라미터 클래스 (Parameterized Class)인 템플릿으로 정의할 수 있다.



[그림 4] 액션 템플릿

액션은 [그림 5]의 액션 (**Action**) 추상 클래스에서 정의된 것과 같이 actionID, actionName, inputValueID, inputValue01, inputValue02, description 등의 속성으로 정의된다. 이 중 actionID, actionName, inputValueID는 정적 속성 (Static Attribute)으로 클래스마다 사전에 결정된 값을 가지고 있어 실제로 시나리오 작성 시 생성하게 되는 액션 객체는 정적 속성 외 값만 결정하게 된다. 예를 들어, **ProgramExecution** 액션 클래스의 경우, actionID는 "101", actionName은 "%1 화면을 구동시킨다", inputValueID는 "00001" (ProgramID)로 결정되어 있다. 이 액션 클래스는 프로그램을 수행하는 작업으로서 입력되는 값은 프로그램ID를 의미한다. 따라서, 수주등록 프로그램을 구동시키기 위한 액션 객체는 inputValue01, inputValue02, description 속성에 대한 값을 객체 속성으로 결정하여 시나리오 작성시 사용된다. 액션 클래스의 `execute()` 연산은 [그림 6]와 같이 액션이 호출되었을 때 시뮬레이션 에이전트가 수행해야 할 코드를 담고 있다.



[그림 5] 액션 클래스와 오브젝트

```
*execute (inputValue01: InputValueType, inputValue02: string): void
{ Window SetContext, "WindowTag=WEBBrowser", ""
Browser SetFrame, "Type=HTMLFrame;HTMLId=frToolbar", ""
EditBox DbClick, "Type=EditBox;Name=btGo", "Coords=60,8"
InputKeys " inputValue01"
HTMLImage Click, "Type=HTMLImage;Name=tbGo", "Coords=20,9"
Delayfor 5000
EditBox DbClick, "Type=EditBox;Name=btGo", "Coords=93,6"
InputKeys "{DELETE}" }
```

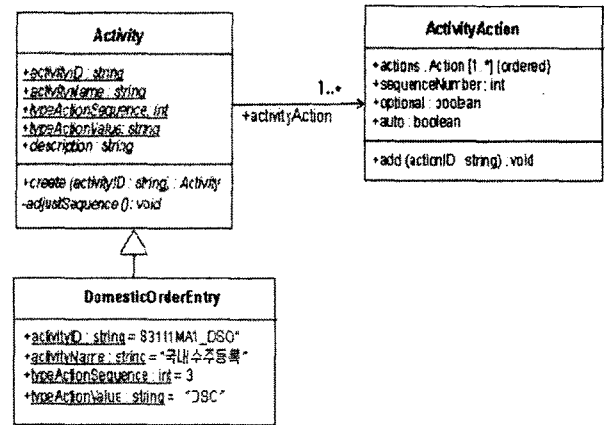
[그림 6] "101" 액션에 대한 시뮬레이션 에이전트 코드

3.3 액티비티 (Activity)

액션은 개별 프로그램을 구동시켜 놓고 마우스나 키보드를 조작하여 GUI를 대상으로 작업하는 것이다. 액션은 순차적으로 조합되어 액티비티로 구성될 때 비로서 의미를 갖는다. 액티비티는 단위 프로그램을 구동하여 수행하게 되는 모든 액션을 조합하여 정의한 것으로 [그림 7]의 액티비티 (**Activity**) 추상 클래스에서 정의된 것과 같이 activityID, activityName, typeActionSequence,

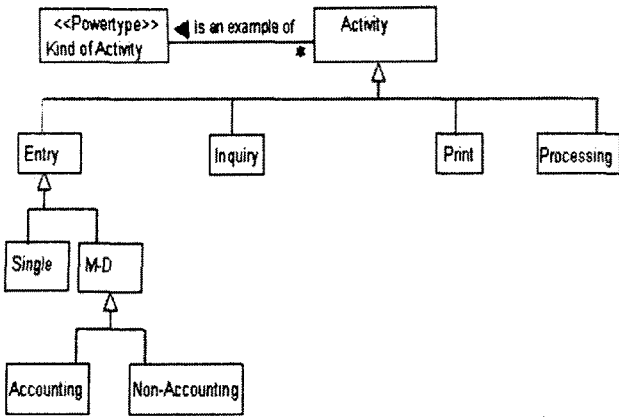
typeActionValue 등의 속성으로 정의된다. 이 중 activityID, activityName, typeActionSequence, typeActionValue 속성은 정적 속성으로 액티비티 클래스마다 사전에 결정된 값을 가지고 있어 실제로 시나리오 작성 시 생성하게 되는 액티비티 객체는 description 속성 값만 결정하게 된다. 예를 들어, **DomesticSalesEntry** 액티비티 클래스의 경우, activityID는 "S3111MA1_DSO", activityName은 "국내수주등록", typeActionSequence는 3, typeActionValue는 "DSO"로 결정되어 있다.

액티비티에서 수행될 액션들을 나타낼 activityAction 속성은 액티비티액션 (**ActivityAction**) 클래스와 관계 (Attributes by Relationship)로 정의되었다. **ActivityAction** 클래스에 의해 정의되는 액션의 컬렉션 (Collection) 객체가 실질적으로 액티비티 객체별 시나리오를 구별하게 된다. 액티비티 내 액션은 반드시 순서에 따라 수행되어야 한다. sequenceNumber는 이와 같이 순번을 지정하기 위한 속성이다.



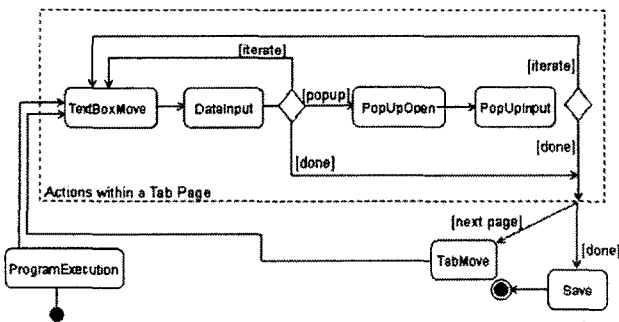
[그림 7] 액티비티 클래스

액티비티 타입은 데이터베이스 작업 유형에 따라 등록, 조회, 출력, 처리 등으로 구분될 수 있다 [그림 8]. 또한, 등록 액티비티는 기본적으로 기존 정보를 등록하거나 단위 거래를 처리하게 되는데 처리 특성에 따라 타입을 분류할 수도 있다. 즉, 등록하게 되는 데이터가 단일 마스터 테이블에 저장되는 경우와 마스터 테이블 (일반 정보)과 디테일 테이블 (내역 정보)에 저장되는 경우를 구분하여 액티비티 타입을 구분할 수도 있다. 일반적으로 마스터-디테일 테이블에 데이터가 저장되는 경우는 거래 처리 (Transaction Processing)의 결과로 볼 수 있다. 이와 같은 거래 처리 결과 회계전표가 자동으로 기표되는 액티비티는 회계적 액티비티, 그렇지 않은 경우는 비회계적 액티비티로 타입을 구분할 수 있다



[그림 8] 액티비티 타입

액티비티의 순차적 액션 진행 상황을 일반적인 액티비티 다이어그램 (Activity Diagram)로 도식화 할 수 있다. [그림 9]는 마스터 테이블 등록 액티비티 타입에 대한 액티비티 다이어그램을 보여 주고 있다. 시나리오 작성은 액티비티 객체에 대한 세부 액션을 새로 구성하는 것이 아니고 사전에 구성된 템플릿에서 필요한 액션을 순차적으로 선택하여 완성할 수 있게 되는 것이다. 따라서, 액션의 선택 여부에 따라 달라 지는 것은 액션의 순번이다. 액티비티 클래스에 정의된 `adjustSequence()` 연산은 템플릿을 사용하여 시나리오를 구성한 후 최종적으로 액션의 순서를 정할 뿐 아니라, 중간 중간에 사용되지 않는 액션으로 인해 다음에 입력할 텍스트박스로 이동하기 위한 탭 키 누르는 횟수를 자동으로 조정하게 된다.

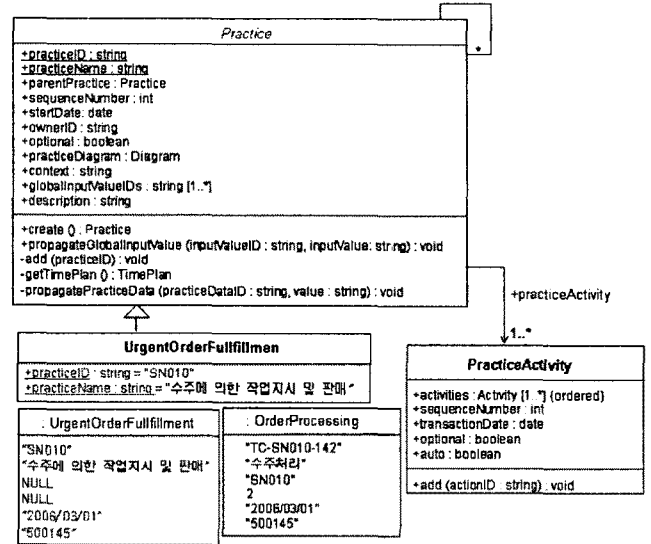


[그림 9] 등록 액티비티의 액티비티 다이어그램

3.4 프랙티스 (Practice)

프랙티스는 [그림 10]의 프랙티스 (**Practice**) 추상 클래스에서 정의된 것과 같이 `practiceID`, `practiceName`, `parentPracticeID`, `sequenceNumber`, `startDate`, `ownerID`, `optional`, `practiceDiagram`, `context`, `globalInputValueIDs` 등의 속성으로 정의된다. 이 중 `practiceID`, `practiceName`, `practiceDiagram`, `Context` 속성은 정적 속성으로 프랙티스 클래스마다 사전에 결정된 값을 가지고 있어 실제로 시나리오 작성 시

생성하게 되는 프랙티스 객체는 클래스 속성 외 값만 결정하게 된다.



[그림 10] 프랙티스 클래스의 객체

3.5 에이전트 (Agents)

시나리오 구성을 지원하기 위한 에이전트는 시나리오 템플릿 구성 에이전트 (Scenario Template Assembling Agent: STAA), 시나리오 구성 지원 에이전트 (Scenario Construction Support Agent: SCSA), 전역 입력값 전파 에이전트 (Global InputValue Propagation Agent: GIPA), 시나리오 시뮬레이션 리코딩 에이전트 (Scenario Simulation Recording Agent: SSRA), 시나리오 검색 지원 에이전트 (Scenario Search Engine Agent: SSEA) 등으로 구성된다.

STAA는 시나리오 작성자가 신규 프랙티스 시나리오를 작성하기 위해 필요한 시나리오 컴포넌트 즉, 액티비티 시나리오 객체 또는 서브프랙티스 시나리오 객체를 선택, 사용 (Pick & Drop) 방식으로 원하는 프랙티스 시나리오의 초기 상태인 템플릿을 작성하는 작업을 지원하는 에이전트이다.

SCSA는 작성된 프랙티스 템플릿으로 실제 프랙티스 사례에 대한 시나리오를 완성할 때 지원하는 에이전트이다.

GIPA는 시나리오 작성자가 프랙티스 템플릿의 각 액션의 입력 데이터를 설정 (Input) 할 때 만일 전역 입력값이 최초로 설정되었다면 같은 입력값을 사용하는 모든 후행 액티비티 템플릿의 액션에 값을 동일하게 설정하는 작업을 수행해 준다. GIPA에 의해 입력값에 대한 시나리오 작성이 용이해 지고 입력 오류도 최소화 할 수 있다.

SSRA는 프랙티스 템플릿에 대한 시나리오가 완성된 후 시나리오에 지정된 대로 각각의 액션에

대한 입력값을 설정하여 일련의 시뮬레이션 소스를 생성하고 소스를 수행하여 가상 시나리오 수행 과정을 녹화하여 저장하는 역할을 담당한다.

SSEA는 기존의 시나리오 베이스에서 유사도에 따라 시나리오를 검색하여 제공하는 역할을 담당한다. 액티비티, 프랙티스 등의 Description 속성을 기준으로 유사도를 계산한다. 검색은 액티비티, 프랙티스 단계별로 세분화되어 진행될 수 있도록 구현할 수 있는 수준 검색 (Level Search) 기법을 적용하였다.

4. 결론

ERP의 성공적인 도입과 운영에 있어 테스트와 교육은 매우 중요하다. 본 논문에서는 ERP시스템 테스트와 교육에 필요한 시나리오를 구축하기 위한 프레임워크인 SF-ETT을 제시하였다. SF-ETT의 기반 구조는 프랙티스로서 ERP 프로세스가 실제 기업에 적용된 사례를 정형화하여 정의하였다. 프랙티스 시나리오를 구성하기 위한 컴포넌트로 액티비티, 액션 클래스가 정의되었으며 이를 사용하여 시나리오를 작성할 때 지원할 수 있는 에이전트를 설계하였다.

SF-ETT는 삼성SDS의 uniERPII 테스트 및 교육 시나리오 작성 프레임워크인 uTF로 구현되고 있다. 현재 57개 프랙티스 시나리오가 작성되었으며 향후 지속적으로 프랙티스를 추가하여 정의하고 시나리오를 작성할 계획이다. uTF는 uniERPII의 버전 업그레이드시 신 버전에 대한 테스트를 자동으로 수행하여 효율적으로 품질 보증 수준으로 향상시키는데 기여하고 있다. 또한, uniERPII 고객사에 커스마이징 작업이 완료되면 uTF를 사용하여 고객사에 맞는 프랙티스별 시나리오를 신속하게 작성하여 제공함으로써 사용자가 실제로 자신이 uniERPII를 사용하여 거래를 처리할 때와 동일한 상황에서의 실습할 수 있도록 지원하고 있다.

SF-ETT의 성공적인 적용은 다음과 같이 보다 효과적인 ERP 시스템의 성공적인 운영에 기여할 수 있을 것으로 기대된다.

- 프랙티스 중심의 보다 효과적인 테스트와 교육이 실현될 수 있다.
- 시나리오 작성 비용을 크게 절감할 수 있다.
- 보다 구체적인 사전 테스트를 통해 ERP 시스템의 품질을 제고할 수 있다.
- 작성된 시나리오에 대한 시뮬레이션을 실행하여 교육 콘텐츠로 활용할 수 있다.

참고 문헌

- Bingi, P, M.K. Sharma and J.K. Godla, "Critical Issues Affecting an ERP Implementation", Information Systems Management, Vol. 16 No. 3 (1999), 7-14.
- Dennis, A., B.H. Wixom and D. Tegarden, Systems Analysis and Design with UML Version 2.0 : An Object-Oriented Approach, 2nd Ed., Hoboken, NJ, John Wiley & Sons, 2004.
- Davenport, T. H., "Putting the enterprise into the enterprise system," Harvard Business Review, Vol. 76, No. 4 (1998), 121-131.
- Gupta, A., "Enterprise Resource Planning: the Emerging Organizational Value Systems," Industrial Management & Data Systems, Vol.100, Iss.3 (2000), 114-118.
- Jacobson, I, et al., Object Oriented Software Engineering, Reading, MA, Addison-Wesley, 1992.
- Kapp, K. M., "The USA Principle: The key to ERP Implementation Success", APICS, June, (1997), 62-66.
- McAlary, S., "Three pitfalls in ERP implementation", Strategy & Leadership, Oct-Dec. Vol. 27, No. 6 (1999), 49.
- 김민수, "통합 비즈니스 프로세스를 통한 기업정보 시스템의 상호운영성", 정보과학회지, 제22권 제7호(2004), 22-25.
- 김훈태, 정한일, 신기태, 임춘성, 시나리오 CBT 기반의 ERP 교육시스템 개발, 1999년도 한국전자거래(CALS EC)학회, 제2권(1999), 597-605.
- 박광호, "객체지향 정보시스템의 테스트를 위한 확장된 유스케이스의 사용과 계층적 상태 기반 테스트 방법", 정보기술과 데이터베이스 저널, 제6권 제2호 (1999), 29-43.
- 박광호, 서형교, "프랙티스 기반 ERP 교육 프레임워크", 2006년 춘계 경영정보학회 학술대회 (2006), 875-882.
- 박광호, 김상수, 김호택, "모기업 주도적 협력업체 정보화 사업 성과에 관한 연구: 삼성전자 사례", Information System Review, 제8권, 제3호 (2006), 227-245.
- 윤중수, 한경구, 한재민, "중소기업 정보화의 주요 관리 이슈와 주요 성공요인에 관한 실증적 연구," 경영학연구, 제27권 제3호(1998), 759-787.
- 이석준, "ERP 시스템 구현의 핵심성공요인과 활용성과에 관한 실증적 연구 : 중소기업을 중심으로," 경영정보학연구, 제11권, 제4호 (2001), 155-174.
- 채정숙, 박종홍, "비즈니스 프로세스 모델링 도구 설계 및 구현", 한국컴퓨터종합학술대회 2005 논문집, 제32권, 제1호 (2005), 166-168.
- 황재호, ERP 시스템의 구축 후 운영 성공에 관한 사례연구, 정보기술과 데이터베이스 저널, 제10권 1호 (2002), 61-70.