

S O F T W A R E E N G I N E E R I N G



부 록 Appendix





Appendix

A

인터넷 쇼핑몰 설계 예제

개요

1. 클래스 다이어그램 CategoryListView | ItemListView | ItemDetailView
2. 시퀀스 다이어그램 상품목록조회 | 상품상세조회
3. 클래스 명세 DAO 클래스 | UI 클래스
4. 설계 클래스 다이어그램
5. 데이터베이스 t_category | t_item
6. 지침 및 고려사항

시스템 설계		프로젝트명: 인터넷 쇼핑몰 시스템 개발	
		시스템명: iShop 인터넷 쇼핑몰	
활동명: 설계	하위 활동명: 기본 설계	작업명: 설계 문서 작성	
문서 번호: CSD_0002	작성자: 이호진	작성일: 2013-08-01	버전: v0.01

개요

이 설계 문서는 인터넷 쇼핑몰 요구사항 명세서 Rev. 2.0을 바탕으로 작성한다. 분석 단계에서 밝혀진 클래스 이외에 사용자 인터페이스를 위한 UI 클래스를 새롭게 식별하여 클래스 다이어그램을 작성하고, 이들 클래스 사이의 상호작용을 나타낸 시퀀스 다이어그램을 작성한다.

다음, 각 클래스에 대한 속성과 오퍼레이션, 각 오퍼레이션에 대한 명세(알고리즘 포함)를 작성한다. 사용자 인터페이스에 요구되는 데이터와 오퍼레이션을 중심으로 화면 설계가 이루어지고, 데이터베이스 설계는 관계형 모델을 사용한다.

이 문서는 인터넷 쇼핑몰의 카테고리화 상품만을 예로 하여 카테고리조회, 상품목록조회, 상품 상세조회에 대한 설계과정을 간단히 설명하고자 한다.

1 클래스 다이어그램

분석 단계에서 도출된 시스템 클래스는 사용자와의 상호작용을 위한 UI를 염두에 두지 않고, 순수한 시스템의 기능을 담당하는 객체로서 식별한다. 분석 단계에서 식별한 클래스들은 그 특성상 시스템의 데이터에 접근하여 조회, 등록, 수정, 삭제 등의 기능을 수행하게 되므로, DAO(Data Access Object)라는 이름을 붙여 정의한다. 설계 단계에서는 프로그램 코드에서 클래스를 식별할 수 있도록 클래스의 이름을 영문으로 바꾸어 정의한다. 분석 단계의 클래스명은 표 1과 같이 변경하여 표시한다.

:: 표 1 | 분석 단계에서 식별된 클래스명과 설계 단계에서의 클래스명

분석 단계 클래스명	설계 단계 클래스명
카테고리	CategoryDAO
상품	ItemDAO

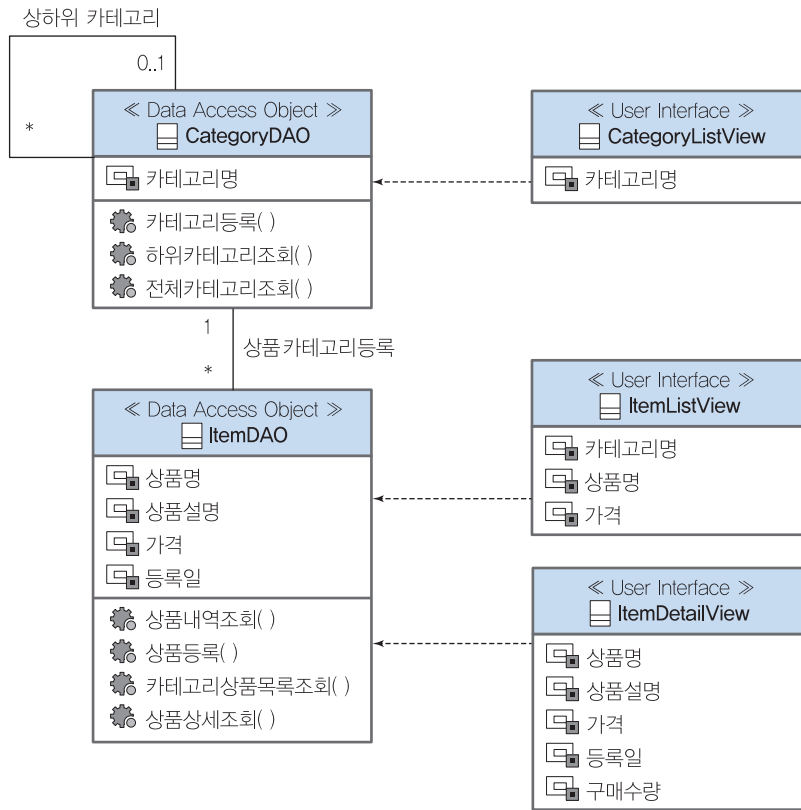
설계 단계에서는 분석 단계에서 정의한 클래스 이외에 유스케이스 시나리오에 기반하여 사용자와의 상호작용을 담당하는 사용자 인터페이스(UI) 클래스를 새롭게 식별한다. UI 클래스는 사용자가 인식하고 이해할 수 있는 GUI를 제공하여 시나리오에 나타난 사용자와 시스템의 상호작용을 처리하는 기능을 담당하며, 사용자에게 보여주는 데이터를 DAO 클래스로부터 전달받거나 사용자로부터 입력받은 데이터를 DAO 클래스에 전달하여 사용자와 시스템 사이의 상호작용을 유연하게 처리하기 위해 정의한다.

UI 클래스를 식별하기 위해서는 먼저 유스케이스 시나리오의 각 항목을 검토하여 시스템이 시나리오에 명시된 기능의 흐름을 제공하기 위해 필요한 사용자 인터페이스를 식별하여, 사용자에게 필요한 정보를 제공하고, 사용자의 입력을 받아들여 처리하는 화면을 하나의 UI 클래스로 정의한다. 유스케이스에서 식별된 UI 클래스는 표 2와 같다.

:: 표 2 | 유스케이스에서 도출된 UI 클래스

유스케이스		시나리오 번호	UI 클래스명	
ID	이름		한글	영문
UC-C02	상품목록조회	기본 1	카테고리조회화면	CategoryListView
		기본 5	상품목록조회화면	ItemListView
UC-C04	상품상세조회	기본 2	상품상세조회화면	ItemDetailView

이상의 내용을 바탕으로 설계 단계에서 새로 작성한 클래스 다이어그램은 다음과 같다. CategoryDAO와 ItemDAO 클래스는 분석 단계에서 작성한 클래스 다이어그램과는 차이가 없으며, 설계 단계에서 식별한 UI 클래스인 CategoryView, ItemListView, ItemDetailView가 추가된 모습을 보여준다. UI 클래스의 속성은 유스케이스를 기반으로 UI 클래스를 식별하는 과정에서 각 UI가 입/출력해야 할 정보를 추출하여 정의한다. 이 다이어그램은 아직 UI 클래스의 오퍼레이션을 나타내지 않고 있는데, 아직 완성된 모습의 다이어그램이 아니기 때문이다. UI 클래스의 오퍼레이션을 식별하기 위해서는 분석 단계에서 작성한 시퀀스 다이어그램을 UI 클래스가 포함된 모습으로 다시 작성하는 과정이 필요하다. 2절에서 설계 단계의 시퀀스 다이어그램을 작성할 것이다.

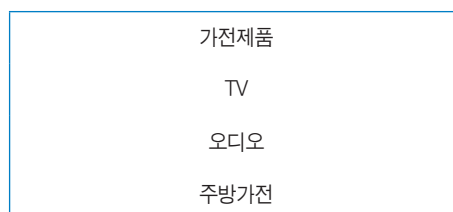


::그림 1 | 설계 클래스 다이어그램(초기)

UI 클래스가 모두 식별되었으면, 유스케이스 시나리오에 기반하여 각 UI의 레이아웃을 설계한다. 유스케이스 시나리오로부터 UI 클래스의 속성을 추출하였으므로, 사용자에게 보여 주어야 할 속성과 보여주지 않아도 될 속성을 결정하여 화면에 출력할 속성을 어떻게 배치할 것인지를 고려하여 레이아웃을 결정해야 한다. 이때 사용자의 편의성을 고려하여 적절한 위치에 필요한 속성을 배치하는 것이 중요하다.

CategoryListView

카테고리의 목록을 출력하는 화면이다. 가전제품의 하위카테고리를 보여주고 있다.



::그림 2 | UI클래스 CategoryListView의 레이아웃

ItemListView

CategoryListView에서 사용자가 최하위 카테고리를 선택한 경우 선택한 카테고리에 속한 상품의 목록을 보여주는 화면이다.

카테고리 : 오디오

▶ 상품 정보

상품명	가격
2채널 스피커	20,000원
3채널 스피커	180,000원
홈시어터	1,264,000원

::그림 3 | UI 클래스 ItemListView의 레이아웃

ItemDetailView

ItemListView에서 사용자가 상품명을 선택한 경우 선택한 상품의 상세정보를 보여주는 화면이다.

▶ 상품 정보

상품명	2채널 스피커
가격	20,000원
등록일자	2013-08-20
상품설명	PC에 사용할 수 있는 스피커
상품사진	

▶ 주문 정보

구매수량	[1]
주문금액	[20,000]

[목록으로] [장바구니담기]

::그림 4 | UI 클래스 ItemDetailView의 레이아웃

2 시퀀스 다이어그램

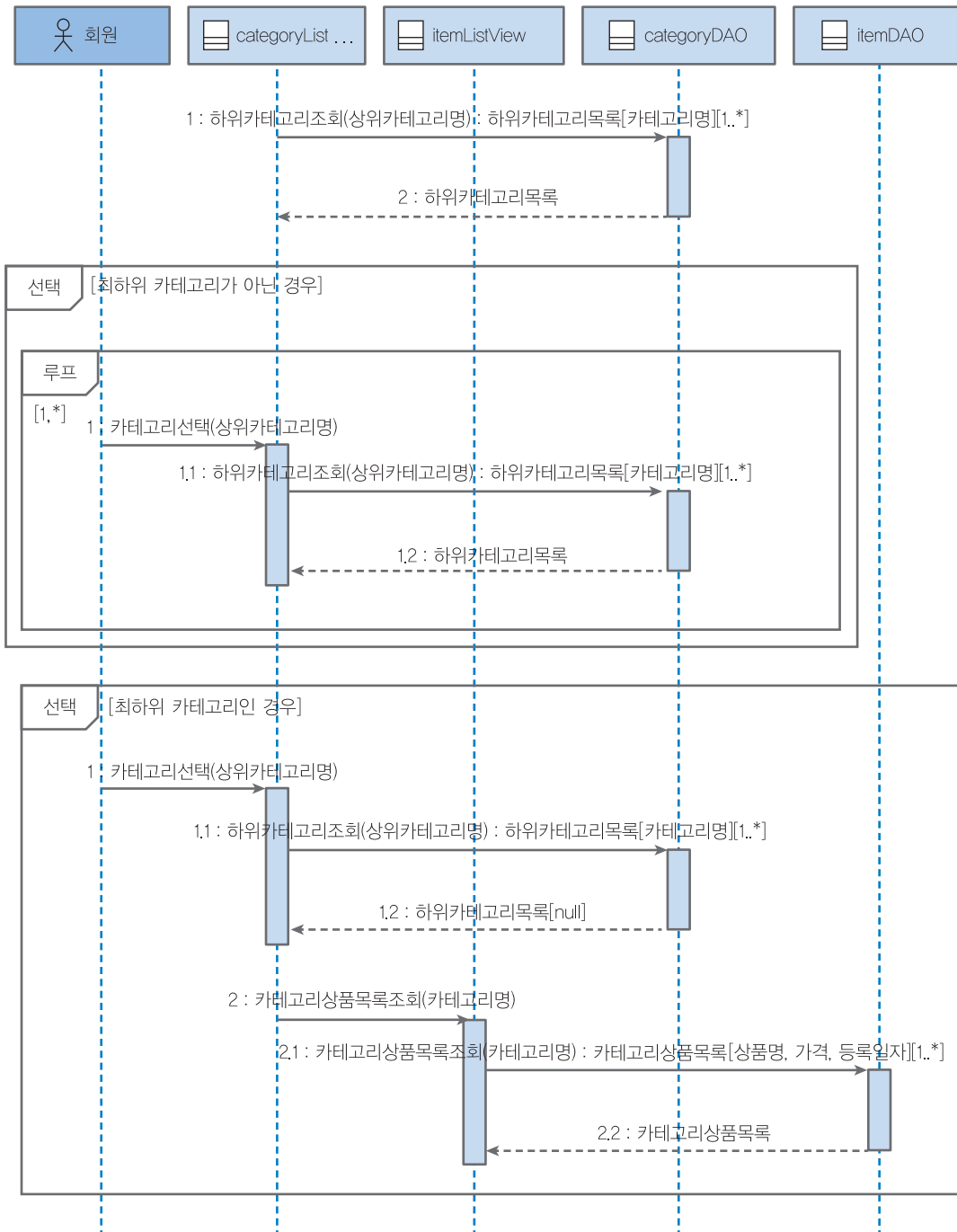
분석 단계에서 각 유스케이스 시나리오별로 작성했던 시퀀스 다이어그램을 확장하여 설계 단계 시퀀스 다이어그램을 작성한다. 이는 설계 단계에서 새롭게 식별된 UI 클래스를 포함한 다이어그램으로 UI 클래스와 DAO 클래스 사이에서 상호작용하는 과정을 나타내기 위해서이다.

다이어그램상의 표기법은 분석 단계에서 작성한 시퀀스 다이어그램과 다르지 않으며, UI 클래스를 포함하여 새로 작성함으로써 사용자-UI 클래스-DAO 클래스 간의 상호작용을 구체적으로 나타낼 수 있다. 사용자가 발생시키는 모든 이벤트는 먼저 UI 클래스에게 전달되며, UI 클래스는 이벤트를 처리하는 과정에서 DAO 클래스에게 또 다른 이벤트를 발생시키는 방식으로 사용자와 DAO 클래스 사이에서 일어나는 상호작용을 원활하게 처리한다.

시퀀스 다이어그램을 작성함으로써 UI 클래스가 사용자의 이벤트를 처리하거나 또 다른 UI 클래스를 호출하는 등의 역할을 하는 오퍼레이션을 식별할 수 있다. 분석 단계에서 작성한 시퀀스 다이어그램과 마찬가지로 설계 단계의 시퀀스 다이어그램에서도 다른 클래스의 오퍼레이션을 호출하는 시점과 이때 필요한 파라미터, 오퍼레이션의 반환값 등의 정보를 정의한다. DAO 클래스의 오퍼레이션은 분석 단계에서 정의한 내용을 기초로 작성하며, UI 클래스의 오퍼레이션을 식별하기 위한 용도로 작성한다.

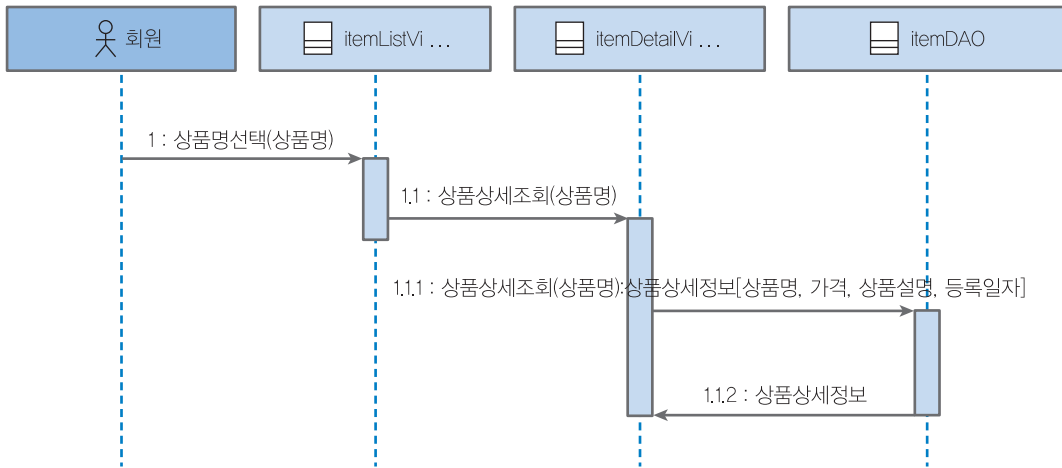
그림 1의 클래스 다이어그램에 나타나 있는 DAO 클래스는 분석 단계에서 식별이 완료된 오퍼레이션을 모두 보여주고 있지만, 여기에서는 두 개의 유스케이스만을 다루기 때문에 DAO 클래스가 가진 오퍼레이션의 일부만이 시퀀스 다이어그램에 나타난다. 설계 단계에서도 클래스의 오퍼레이션은 시퀀스 다이어그램 작성을 통해 식별되므로, 여기에서 다루지 않는 나머지 유스케이스에 대하여 시퀀스 다이어그램을 작성해 보면 그림 1 클래스 다이어그램의 모든 오퍼레이션을 구할 수 있을 것이다.

상품목록조회



::그림 5 | 상품목록조회 시퀀스 다이어그램

상품상세조회



::그림 6 | 상품상세조회 시퀀스 다이어그램

③ 클래스 명세

UI 클래스가 포함된 시퀀스 다이어그램을 작성하여 UI 클래스의 오퍼레이션이 식별되었으므로, DAO 클래스와 UI 클래스의 명세서를 작성한다.

DAO 클래스

(1) CategoryDAO

① 개요

카테고리에 대한 정보와 카테고리의 등록, 수정, 삭제 및 조회 기능을 포함하는 클래스

② 관련 UI 클래스

CategoryListView

③ 속성

속성명	타입	논리 모델 속성명	설명
ctgrName	String	카테고리명	카테고리의 이름
upCtgrName	String	상위카테고리명	해당 카테고리가 속한 상위카테고리의 이름

④ 오퍼레이션 목록

이름	카테고리등록
영문명	createCategory
설명	신규 카테고리를 등록한다.
파라미터	1. p_ctgr_name(카테고리명) 2. p_up_ctgr_name(상위카테고리명)
반환값 및 타입	저장수행결과(String) <ul style="list-style-type: none"> • 'Y': 카테고리가 정상적으로 저장된 경우 • 'N': 카테고리 저장 도중 오류가 발생한 경우 • 카테고리명: 카테고리명이 중복되어 저장할 수 없는 경우
관련 테이블	t_category
개요	동일한 이름의 카테고리가 2개 이상 존재할 수 없다. 따라서 저장하기 전에 파라미터로 입력받은 카테고리명과 동일한 카테고리명을 갖는 카테고리의 존재 여부를 검사하여, 동일한 카테고리가 존재하는 경우 DB에 저장하지 않는다. 유일한 카테고리임이 확인되면 DB에 저장하고 그 결과를 반환한다.
알고리즘	<pre> if 입력받은 카테고리명과 동일한 카테고리가 이미 존재 카테고리를 생성하지 않는다. return FALSE else 상위카테고리의 아래에 새로운 카테고리를 생성한다. end if return 저장수행결과 [사용 SQL] INSERT INTO t_category (ctgr_name, up_ctgr_name) VALUES (p_ctgr_name, p_up_ctgr_name) </pre>

이름	하위카테고리조회
영문명	getSubCategory
설명	1. 상위카테고리명을 파라미터로 입력받는다. 2. 파라미터인 상위카테고리명의 하위카테고리목록을 DB에서 조회한다. 3. 조회한 결과를 반환한다.
파라미터	1. p_up_ctgr_name(상위카테고리명)
반환값 및 타입	하위카테고리목록(Vector)
관련 테이블	t_category
개요	파라미터로 입력받은 상위카테고리의 바로 하위 레벨에 속한 카테고리목록을 조회한다. 오퍼레이션 getCategoryTree와는 달리 바로 다음 레벨의 카테고리만을 조회한다.
알고리즘	[사용 SQL] SELECT ctgr_name FROM t_category WHERE up_ctgr_name = p_up_ctgr_name

이름	전체카테고리조회
영문명	getCategoryTree
설명	선택한 카테고리의 모든 하위 레벨에 속한 카테고리의 목록을 조회한다.
파라미터	1. p_ctgr_name(카테고리명)
반환값 및 타입	전체카테고리목록(Vector)
관련 테이블	t_category
개요	파라미터로 입력받은 카테고리의 모든 하위 레벨에 속한 카테고리목록을 조회한다. Root 노드의 카테고리명을 파라미터로 사용해 호출하여 전체카테고리의 목록을 조회하는 데 사용한다.
알고리즘	[사용 변수] • 하위카테고리목록(Vector) 지정한 카테고리의 바로 아래 하위카테고리목록을 조회한 결과 • 반환값(Vector) getCategoryTree()의 반환값 지정한 카테고리의 모든 하위카테고리의 목록 하위에 속한 모든 카테고리의 목록을 저장한다. 하위카테고리의 단계에 따라 들어쓰기가 포함된 결과 파라미터로 입력받은 카테고리의 모든 하위 레벨에 속한 카테고리의 목록이 저장된다. 재귀호출을 통해 반복되며 하위카테고리의 반환값이 그 상위카테고리에 전달되며, 상위카테고리에서는 모든 하위카테고리들의 반환값을 연결하여 자신의 반환값을 만들어 반환한다. 예) 카테고리 A의 하위에 A1, A2가 속해 있고, A1의 하위에 A11, A12가 속해 있는 경우 반환값에는 [1] A [2] A1 [3] A11 [4] A12 [5] A2 의 형태로 트리 구조로 화면에 출력하기 위한 모양을 갖추어 저장된다([1]~[5]는 index를 의미한다).

알고리즘

```

• 임시반환값(Vector)
getCategoryTree() 재귀호출의 하위단계에서 반환한 값
위의 예에서 파라미터를 A로 갖는 getCategoryTree('A')에서 하위카테고리 A1에 대해 getCategoryTree('A1')을
재귀호출한 결과는
[1] A1
[2] A11
[3] A12
의 형태를 갖게 된다.

[알고리즘]
# getCategory()를 호출하여 하위카테고리목록을 조회한다.
set 하위카테고리목록 to getCategory(카테고리명);

# 반환값의 마지막에 카테고리명을 더한다.
set 반환값 to 반환값 + 카테고리명;

# LOOP: 하위카테고리의 모든 카테고리에 대해
for i := 1 to 하위카테고리목록의 size do

    # getCategoryTree()를 재귀호출하고 결과를 임시반환값에 저장
    # 하위카테고리목록[i]는 하위카테고리목록의 i번째 element
    set 임시반환값 to getCategoryTree(하위카테고리목록[i]);
    # LOOP: 각 하위카테고리에 속한 모든 카테고리의 목록에 대해
    for j := 1 to 임시반환값의 size do

        # 들여쓰기를 위해 앞에 공백을 붙여 반환값에 더한다.
        # recursion이 반복될수록 공백이 중첩되므로 카테고리 레벨이 낮아질수록 더 많은 공백 문자가 카테
        고리명 앞에 붙게 된다. 결과를 트리 형태로 보이게끔 하기 위해(공백의 수는 화면 layout에 따라 임의
        로 조절)
        # 임시반환값[j]는 임시반환값의 j번째 element
        set 반환값 to 반환값 + " " + 임시반환값[j];
    end
end

# 반환값을 반환한다.

return 반환값

```

(2) ItemDAO

① 개요

상품에 대한 정보와 상품의 등록, 수정, 삭제 및 조회 기능을 포함하는 클래스

② 관련 UI 클래스

ItemListView, ItemDetailView

③ 속성

속성명	타입	논리 모델 속성명	설명
itemName	String	상품명	상품의 이름
ctgrName	String	카테고리명	해당 상품이 속해 있는 카테고리의 이름
description	String	상품설명	상품에 대한 상세정보를 기술한 속성
itemImg	Image	상품사진	상품의 사진이 저장된 경로
cost	Int	가격	상품의 가격
regDate	String	등록일	상품을 온라인 쇼핑몰 시스템에 등록한 날짜

④ 오퍼레이션 목록

이름	상품내역조회
영문명	getItemInfo
설명	입력받은 상품의 가격정보를 조회하여 반환한다. 1. 입력받은 상품명에 해당하는 상품 가격의 목록을 DB에서 조회한다. 2. 조회한 결과를 반환한다.
파라미터	1. p_item_list(상품명)
반환값 및 타입	가격정보가 포함된 상품목록(Vector)
관련 테이블	t_item
개요	파라미터로 입력받은 상품의 가격정보를 조회한 결과를 반환한다.
알고리즘	[사용 SQL] SELECT item_name, cost FROM t_item WHERE item_name IN (p_item_list)

이름	상품등록
영문명	createItem
설명	상품을 등록한다.
파라미터	1. p_item_name(상품명) 2. p_ctgr_name(카테고리명) 3. p_description(상품설명) 4. p_item_img(상품사진) 5. p_cost(가격)
반환값 및 타입	상품저장결과(String) • ‘Y’: 상품 저장이 정상적으로 실행된 경우 • ‘N’: 상품 저장 도중 오류가 발생한 경우 • 상품명: 상품명에 중복되어 저장할 수 없는 경우
관련 테이블	t_item

개요	동일한 이름의 상품이 2개 이상 존재할 수 없다. 따라서 저장하기 전에 파라미터로 입력받은 상품명과 동일한 상품명에 갖는 상품의 존재 여부를 검사하여, 동일한 상품이 존재하는 경우 DB에 저장하지 않는다. 유일한 상품임이 확인되면 DB에 저장하고 그 결과를 반환한다.
알고리즘	<pre> if 입력받은 상품명과 동일한 상품이 이미 존재 return FALSE else 상품 데이터를 생성한다. end if return 상품저장결과 [사용 SQL] INSERT INTO t_item(item_name, ctgr_name, description, item_img, cost) VALUES (p_item_name, p_ctgr_name, p_description, p_item_img, p_cost) </pre>

이름	카테고리상품목록조회
영문명	getItemList
설명	<p>상품목록을 조회한다.</p> <ol style="list-style-type: none"> 1. 카테고리명을 입력받는다. 2. 입력받은 카테고리에 속하는 상품의 목록을 DB에서 조회한다. 3. 조회한 결과를 반환한다.
파라미터	1. p_ctgr_name(카테고리명)
반환값 및 타입	카테고리에 속한 상품목록(Vector)
관련 테이블	t_item
개요	파라미터로 입력받은 카테고리에 속한 상품의 목록을 조회한 결과를 반환한다.
알고리즘	<pre> [사용 SQL] SELECT item_name, cost, reg_date FROM t_item WHERE ctgr_name = p_ctgr_name </pre>

이름	상품상세조회
영문명	getItemDetail
설명	상품상세정보를 조회한다. 1. 상품명을 입력받는다. 2. 입력받은 상품명에 갖는 상품의 상세정보를 DB에서 조회한다. 3. 조회한 결과를 반환한다.
파라미터	1. p_item_name(상품명)
반환값 및 타입	상품정보(ItemDAO)
관련 테이블	t_item
개요	파라미터로 입력받은 상품의 상세정보를 조회한 결과를 반환한다.
알고리즘	[사용 SQL] SELECT item_name, description, item_img, cost, reg_date FROM t_item WHERE item_name = p_item_name

UI 클래스

(1) CategoryListView

클래스명	CategoryListView		
화면명	카테고리조회화면		
화면 개요	카테고리목록을 조회한다.		
관련 테이블	카테고리(t_category)	관련 DAO 클래스	CategoryDAO
관련 유스케이스	UC-C02 상품목록조회		

① 화면

가전제품
TV
오디오
주방가전

② 처리 개요

최초 호출 시 최상위 카테고리 목록을 조회한다. 사용자가 카테고리를 선택하면 선택한 카테고리의 하위카테고리목록이 조회되며, 하위카테고리가 존재하지 않으면 카테고리에 속한 상품목록조회 화면(ItemListView)을 호출한다.

③ 화면 입/출력 정보 일람

번호	I/O	한글명	영문명	자료 형태
1	출력	카테고리명	ctgr_name	char

④ 오퍼레이션 목록

이름	카테고리선택			
영문명	select_category			
파라미터	카테고리명(ctgr_name)			
반환값	하위카테고리의 목록(카테고리명[1..*])			
개요	카테고리명 클릭 이벤트 핸들러			
호출 시점	사용자 이벤트: 카테고리명 선택			
동작	1. 하위카테고리목록을 조회한다. 2a. 하위카테고리목록조회 결과가 null이면 - 상품목록조회화면(ItemListView) 호출 2b. 하위카테고리목록조회 결과가 null이 아니면 - 하위카테고리목록조회 결과를 출력한다.			
호출 DAO		파라미터		
CategoryDAO.하위카테고리조회		카테고리명		
호출 UI		파라미터		
ItemListView.카테고리상품목록조회		카테고리명		

(2) ItemListView

클래스명	ItemListView		
화면명	상품목록조회화면		
화면 개요	카테고리에 해당하는 상품의 목록을 조회한다.		
관련 테이블	상품(t_item)	관련 DAO 클래스	ItemDAO
관련 유스케이스	UC-C02 상품목록조회		

① 화면

카테고리 : 오디오

▶ 상품 정보

상품명	가격
2채널 스피커	20,000원
3채널 스피커	180,000원
홈시어터	1,264,000원

② 처리 개요

카테고리에 속한 상품의 목록을 조회한다. 사용자가 상품명을 선택하면 상품상세조회화면(ItemDetailView)으로 이동한다.

③ 화면 입/출력 정보 일람

번호	I/O	한글명	영문명	자료 형태
1	입력/출력	카테고리명	ctgr_name	char
2	출력	상품명	item_name	char
3	출력	가격	cost	decimal

④ 오퍼레이션 목록

이름	카테고리상품목록조회			
영문명	get_item_list			
파라미터	카테고리명(ctgr_name)			
반환값	카테고리상품목록(상품명, 가격)			
개요	카테고리에 속한 상품의 목록을 조회한다.			
호출 시점	카테고리조회화면(CategoryListView)의 오퍼레이션 카테고리 선택에서 최하위 카테고리가 선택된 경우			
동작	카테고리에 속한 상품목록을 조회하여 출력한다.			
호출 DAO		파라미터		
ItemDAO, 카테고리상품목록조회		카테고리명		
호출 UI		파라미터		
-		-		

이름	상품명 선택
영문명	select_item_name

파라미터	상품명(item_name)
반환값	없음
개요	상품상세정보 UI를 호출한다.
호출 시점	사용자 이벤트: 상품명 선택
동작	상품의 상세정보를 조회하는 상품상세정보화면(ItemDetailView)을 호출한다.
호출 DAO	파라미터
-	-
호출 UI	파라미터
ItemDetailView.상품상세조회	상품명

(3) ItemDetailView

클래스명	ItemDetailView		
화면명	상품상세조회화면		
화면 개요	상품의 상세정보를 조회한다.		
관련 테이블	상품(t_item)	관련 DAO 클래스	ItemDAO
관련 유스케이스	UC-C04 상품상세조회		

① 화면

▶ 상품 정보

상품명	2채널 스피커
가격	20,000원
등록일자	2013-08-20
상품설명	PC에 사용할 수 있는 스피커
상품사진	

▶ 주문 정보

구매수량	[1]
주문금액	[20,000]

[목록으로] [장바구니담기]

② 처리 개요

상품에 대한 상세정보(상품명, 가격, 등록일자, 상품설명, 상품사진)를 조회한다. 상품을 장바구니에 담기 위해 구매수량을 선택할 수 있다. 상품의 수량을 입력한 후 장바구니담기 기능을 실행할 수 있다.

③ 화면 입/출력 정보 일람

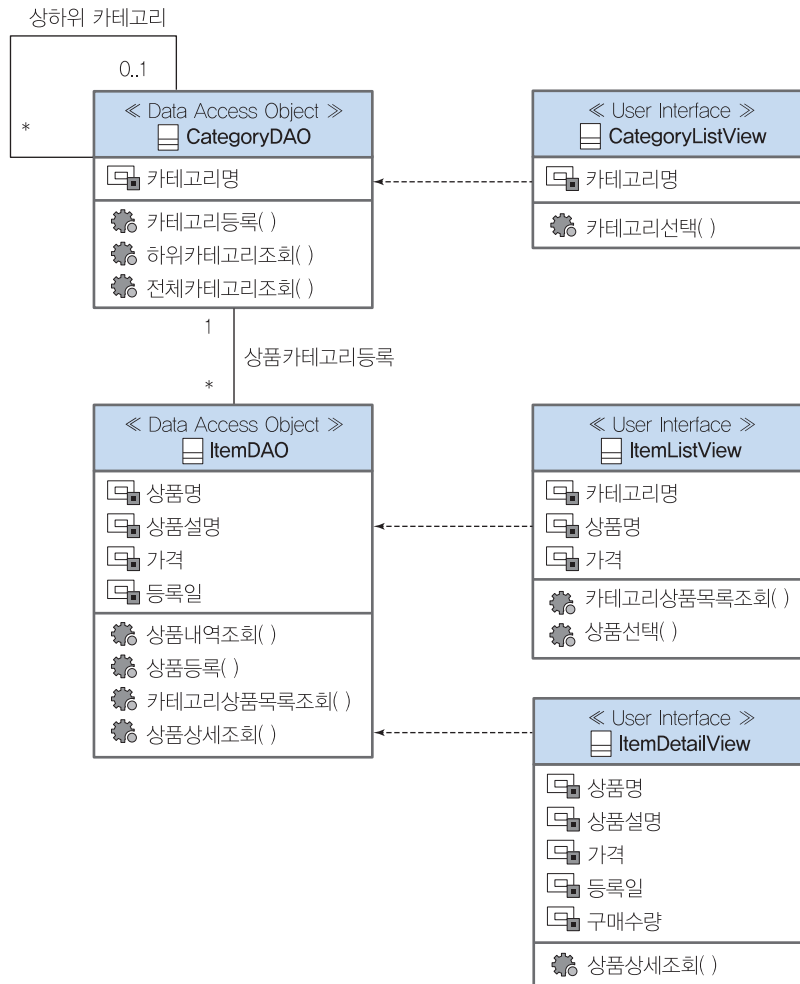
번호	I/O	한글명	영문명	자료 형태
1	입력	구매수량	qty	decimal
2	출력	상품명	item_name	char
3	출력	가격	cost	char
4	출력	등록일자	reg_date	char
5	출력	상품설명	description	char
6	출력	상품사진	item_img	char
7	출력	주문금액	amt	decimal

④ 오퍼레이션 목록

이름	상품상세조회		
영문명	get_item_detail		
파라미터	상품명(item_name)		
반환값	상품의 상세정보(상품명, 가격, 등록일자, 상품설명, 상품사진)		
개요	상품목록조회화면(ItemListView)에서 전달받은 상품명에 해당하는 상품의 상세정보를 조회한다.		
호출 시점	상품목록조회화면(ItemListView)의 오퍼레이션 상품명 선택에서 호출		
동작	상품의 상세정보를 조회하여 출력한다.		
호출 DAO		파라미터	
ItemDAO.상품상세조회		상품명	
호출 UI		파라미터	
-		-	

4 설계 클래스 다이어그램

설계 단계가 완료되어 최종적으로 완성된 클래스 다이어그램의 모습은 다음과 같다. 그림 1과 비교하여 UI 클래스의 오퍼레이션이 추가되어 완성된 모습이다.



::그림 7 | 설계 클래스 다이어그램(최종)

5 데이터베이스

데이터베이스 설계에서는 각각의 테이블에 대한 물리적인 상세 정보를 작성한다. 분석 단계의 클래스 다이어그램에서 각 클래스 간의 관계가 밝혀졌으므로, 이 단계에서는 각 클래스가 가진 속성에 덧붙여 클래스 간의 관계에 의해 식별된 속성을 포함하여 데이터베이스 테이블을 설계한다. 일대일, 일대다 관계에 의해 생성된 속성은 테이블에서 외래 키(FK: Foreign Key)로 나타낸다.

각각의 테이블은 분석 단계에서 식별된 클래스를 나타내며, 이는 결국 설계 단계에서의 DAO 클래스를 의미한다. 분석 단계와 설계 단계에서 정의한 클래스의 이름과 데이터베이스 테이블의 이름 사이의 관계는 표 3과 같다.

::표 3 | 분석-설계 단계 클래스명과 데이터베이스 테이블명의 관계

분석 단계 클래스명	설계 단계 클래스명	테이블명
카테고리	CategoryDAO	t_category
상품	ItemDAO	t_item

t_category

카테고리 클래스의 속성을 저장하는 테이블이다. 상위-하위 카테고리 간의 관계를 나타내기 위한 자기 참조 관계가 존재하며, 이를 위해 자신의 PK를 참조하는 FK인 상위카테고리명 컬럼이 포함되었다. 이때 상위카테고리명이 null인 레코드는 최상위 카테고리를 의미한다.

테이블명	t_category					
순번	한글명	영문명	데이터 타입	크기	NULL 허용	PK/FK
1	카테고리명	ctgr_name	char	50		PK
2	상위카테고리명	up_ctgr_name	char	50	O	FK

t_item

상품 클래스의 속성을 저장하는 테이블이다. 카테고리에 상품이 속한 관계를 나타내기 위해 카테고리명이 FK로 포함되었다.

테이블명	t_item					
순번	한글명	영문명	데이터 타입	크기	NULL 허용	PK/FK
1	상품명	item_name	char	50		PK
2	상품설명	description	char	500		
3	상품사진	item_image	char	200	O	
4	가격	cost	decimal	9		
5	등록일자	reg_date	char	8		
6	카테고리명	ctgr_name	char	50		FK

6 지침 및 고려사항

설계 단계는 기본 설계와 상세 설계로 구분할 수 있다. 기본 설계 단계에서는 UI 설계, 시퀀스 다이어그램 작성, 클래스 명세서 작성 및 데이터 설계가 이루어지며, 상세 설계 단계에서 각 오퍼레이션에 대한 알고리즘 작성(프로시저 설계)이 이루어진다. 또한 기본 설계는 구현 환경의 특정 제품과는 독립적으로 최적의 논리적 해결 방안을 찾는 데 비해, 상세 설계는 기본 설계 단계에서 정의된 최적의 논리적인 방안이 구현/사용상의 편리성이나 성능을 위해 시스템 운영 환경에 사용될 특정 소프트웨어 제품에 따라 다소 조정될 수 있다.

설계 단계 활동이 완료되면, 설계 단계에서 도출된 프로그램 구조 및 명세와 데이터베이스 구조는 소프트웨어 개발의 다음 단계인 구현 단계에서 실제로 프로그래밍 언어를 사용하여 작성된다. 따라서 설계 단계에서는 구현 단계에서 프로그램을 작성하는 데 필요한 정보를 제공해야 할 필요가 있다.



Appendix

B

인터넷 쇼핑몰 코드 예제

코드 예제

1. CategoryDAO.java
2. ItemDAO.java

데이터베이스 생성

코드 예제

설계 문서에 따라 직접 작성한 프로그램의 예제를 첨부한다. 예제 코드는 설계 단계 산출물인 클래스 명세서를 바탕으로 DAO 클래스를 구현한 것이다. 프로그램은 Java를 사용하여 작성되었으나, 다른 객체지향 프로그래밍 언어를 사용하더라도 구현에 큰 차이는 없으리라 생각한다.

UI 클래스의 경우는 사용 언어 및 구현 환경에 따라 코드의 많은 부분이 달라질 수 있기 때문에 예제에는 포함하지 않았다.



주의

이 예제 코드는 책에서 소개되지 않은 독자적인 라이브러리를 사용하였으므로 일반적인 프로그래밍 환경에서 그대로 사용하기는 어려울 것이다. 여기에서는 설계 단계를 거쳐 구현된 프로그램의 모습을 보여주기 위해 참고할 목적으로 소개하는 것이므로, 이 점을 유의하여 참고해 주기 바란다.

1 CategoryDAO.java

```
// 클래스: 카테고리(CategoryDAO)
// 개요: 카테고리에 대한 정보와 카테고리의 등록, 수정, 삭제 및 조회 기능을 포함한다.
public class CategoryDAO {
    private String ctgrName; // 속성: 카테고리명
    private String upCtgrName; // 속성: 상위카테고리명

    // 오퍼레이션: 카테고리등록
    // 파라미터: 카테고리명(String), 상위카테고리명(String)
    // 반환값: 저장수행결과(String)
    public String createCategory(String ctgr_name, String up_ctgr_name) throws Exception
    {
        // 데이터베이스에 접근하고, 쿼리 수행 결과를 저장하기 위한 변수 선언
        DBConnectionPoolMgr dbMgr = null;
        Connection conn = null;
        Statement stmt = null;

        // 반환값 변수 선언
        // 'Y': 저장 성공
        // 'N': 저장 오류
    }
}
```

```

// 카테고리명: 카테고리명 중복
String returnStr = new String();
int res = -1;

try {
    // 데이터베이스에 접근하기 위한 라이브러리에서 사용할 변수 설정
    dbMgr = DBConnectionPoolMgr.getInstance();
    conn = dbMgr.getConnection();
    stmt = conn.createStatement();

    // 쿼리 문자열 변수 선언
    StringBuffer query = new StringBuffer();
    StringBuffer cQuery = new StringBuffer();

    // 상품명 중복 여부 조회 쿼리 문자열 생성
    query.append("SELECT ISNULL(MAX('Y'),'N') AS ctgr_exist_yn");
    query.append("FROM t_category");
    query.append("WHERE ctgr_name = " + ctgr_name + "");

    // 쿼리를 데이터베이스에 전달하여 실행
    ResultSet rs = stmt.executeQuery(query.toString());

    // 카테고리명 중복 여부 쿼리 수행 결과를 변수에 저장
    rs.next();
    String ctgrExistYn = rs.getString("ctgr_exist_yn");

    // 카테고리명이 중복되는 경우
    if (ctgrExistYn.equals("Y")) {
        returnStr = ctgr_name;
    }
    // 카테고리명이 중복되지 않는 경우
    else {
        // 신규 카테고리 정보 저장 쿼리 문자열 생성
        cQuery.append("INSERT INTO t_category(ctgr_name, up_ctgr_name)");
        cQuery.append("VALUES (" + ctgr_name + ", " + up_ctgr_name + ")");

        // 쿼리를 데이터베이스에 전달하여 실행
        res = stmt.executeUpdate(cQuery.toString());

        // 쿼리 실행 중 오류 여부에 따른 반환값 설정
        if (res == 1) returnStr = "Y";
        else returnStr = "N";
    }

    return returnStr;

} catch (Exception e) {
    throw new Exception(e.getMessage());
}
}

```

```

// 오퍼레이션: 하위카테고리조회
// 파라미터: 상위카테고리명(String)
// 반환값: 하위카테고리목록(Vector)
public Vector getSubCategory(String up_ctgr_name) throws Exception
{
    // 데이터베이스에 접근하고, 쿼리 수행 결과를 저장하기 위한 변수 선언
    DBConnectionPoolMgr dbMgr = null;
    Connection conn = null;
    Statement stmt = null;
    ResultSet query_result = null;

    // 반환값 변수 선언
    Vector return_vector = new Vector();

    try {
        // 데이터베이스에 접근하기 위한 라이브러리에서 사용할 변수 설정
        dbMgr = DBConnectionPoolMgr.getInstance();
        conn = dbMgr.getConnection();
        stmt = conn.createStatement();

        // 쿼리 문자열 생성
        StringBuffer query = new StringBuffer();
        query.append("SELECT ctgr_name");
        query.append("FROM t_category");
        query.append("WHERE up_ctgr_name = " + up_ctgr_name + "");

        // 쿼리를 데이터베이스에 전달하여 실행
        query_result = stmt.executeQuery(query.toString());

        // 쿼리 수행 결과를 반환값 변수에 저장
        while (query_result.next()) {
            CategoryDAO category_list = new CategoryDAO();
            category_list.ctgrName = query_result.getString("ctgr_name");
            return_vector.add(category_list);
        }

        return return_vector;
    } catch (Exception e) {
        throw new Exception(e.getMessage());
    }
}

// 오퍼레이션: 전체카테고리조회
// 파라미터: 카테고리명(String)
// 반환값: 전체카테고리목록(Vector)
public Vector getCategoryTree(String ctgr_name) throws Exception
{

```

```
Vector return_vector = new Vector(); // 반환값
Vector sub_result = new Vector(); // 하위카테고리조회 결과
Vector temp_result = new Vector(); // 임시반환값

try {
    // 하위카테고리목록 조회
    sub_result = getSubCategory(ctgr_name);

    // 카테고리명이 null이 아니면 반환값에 카테고리명을 덧붙인다.
    if (!ctgr_name.equals(""))
        return_vector.add(ctgr_name);

    // 하위카테고리목록 type 변환(Vector -> CategoryDAO[])
    CategoryDAO[] sub_data = new CategoryDAO[sub_result.size()];
    sub_result.copyInto(sub_data);

    // loop: 모든 하위카테고리에 대해
    for (int idx=0; idx<sub_data.length; idx++) {
        // 재귀호출한 결과를 임시반환값에 저장
        // 임시반환값 = 하위의 모든 카테고리 목록
        temp_result = getCategoryTree(sub_data[idx].ctgrName);

        // 임시반환값 type 변환 (Vector -> String[])
        String[] temp_data = new String[temp_result.size()];
        temp_result.copyInto(temp_data);

        // loop: 모든 하위카테고리에 대해
        // 모든 하위카테고리의 항목마다 문자열 앞에 공백을 붙이고, 반환값에 카테고리명을 덧붙인다.
        for (int j=0; j<temp_data.length; j++) {
            return_vector.add("                    " + temp_data[j]);
        }
    }

return return_vector;

} catch(Exception e) {
    throw new Exception(e.getMessage());
}
```

2 ItemDAO.java

```
// 클래스: 상품(ItemDAO)
// 개요: 상품에 대한 정보와 상품의 등록, 수정, 삭제 및 조회와 같은 기능을 포함한다.
public class ItemDAO {
```

```

private String itemName; // 속성: 상품명
private String ctgrName; // 속성: 카테고리명
private String description; // 속성: 상품설명
private String itemImg; // 속성: 상품사진
private int cost; // 속성: 가격
private String regDate; // 속성: 등록일

// 오퍼레이션: 상품내역조회
// 파라미터: 상품명(String)
// 반환값: 가격정보가 포함된 상품목록(Vector)
public Vector getItemInfo(String item_list) throws Exception
{
    // 데이터베이스에 접근하고, 쿼리 수행 결과를 저장하기 위한 변수 선언
    DBConnectionPoolMgr dbMgr = null;
    Connection conn = null;
    Statement stmt = null;
    ResultSet query_result = null;

    // 반환값 변수 선언
    Vector return_vector = new Vector();

    try {
        // 데이터베이스에 접근하기 위한 라이브러리에서 사용할 변수 설정
        dbMgr = DBConnectionPoolMgr.getInstance();
        conn = dbMgr.getConnection();
        stmt = conn.createStatement();

        // 쿼리 문자열 생성
        StringBuffer query = new StringBuffer();
        query.append("SELECT item_name, cost");
        query.append("FROM t_item");
        query.append("WHERE item_name IN (" + item_list + ")");

        // 쿼리를 데이터베이스에 전달하여 실행
        query_result = stmt.executeQuery(query.toString());

        // 쿼리 수행 결과를 반환값 변수에 저장
        while (query_result.next()) {
            BucketDAO bucket_list = new BucketDAO();
            bucket_list.itemName = query_result.getString("item_name");
            bucket_list.cost = query_result.getInt("cost");

            return_vector.add(bucket_list);
        }

        return return_vector;
    } catch (Exception e) {

```

```

        throw new Exception(e.getMessage());
    }
}

// 오퍼레이션: 상품등록
// 파라미터: 상품명(String), 카테고리명(String), 상품설명(String), 상품사진(String), 가격(String)
// 반환값: 상품저장결과(String)
public String createItem(String item_name, String ctgr_name, String cost, String description)
throws Exception
{
    // 데이터베이스에 접근하고, 쿼리 수행 결과를 저장하기 위한 변수 선언
    DBConnectionPoolMgr dbMgr = null;
    Connection conn = null;
    Statement stmt = null;

    // 반환값 변수 선언
    // 'Y': 저장 성공
    // 'N': 저장 오류
    // 상품명: 상품명 중복
    String returnStr = new String();
    int res = -1;

    try {
        // 데이터베이스에 접근하기 위한 라이브러리에서 사용할 변수 설정
        dbMgr = DBConnectionPoolMgr.getInstance();
        conn = dbMgr.getConnection();
        stmt = conn.createStatement();

        // 쿼리 문자열 변수 선언
        StringBuffer query = new StringBuffer();
        StringBuffer cQuery = new StringBuffer();

        // 현재 날짜
        java.util.Date today = new java.util.Date();
        SimpleDateFormat simpleDate = new SimpleDateFormat("yyyyMMdd");
        String str_today = simpleDate.format(today);

        // 상품명 중복 여부 조회 쿼리 문자열 생성
        query.append("SELECT ISNULL(MAX('Y'),'N') AS item_exist_yn");
        query.append("FROM t_item");
        query.append("WHERE item_name = " + item_name + "");

        // 쿼리를 데이터베이스에 전달하여 실행
        ResultSet rs = stmt.executeQuery(query.toString());

        // 상품명 중복 여부 쿼리 수행 결과를 변수에 저장
        rs.next();
        String itemExistYn = rs.getString("item_exist_yn");

```

```

// 상품명 중복되는 경우
if (itemExistYn.equals("Y")) {
    returnStr = item_name;
}
// 상품명 중복되지 않는 경우
else {
    // 신규 상품정보 저장 쿼리 문자열 생성
    cQuery.append("INSERT INTO t_item(item_name, ctgr_name, cost, description, reg_date)");
    cQuery.append("VALUES (" + item_name + "");
    cQuery.append(" , " + ctgr_name + "");
    cQuery.append(" , " + cost + "");
    cQuery.append(" , " + description + "");
    cQuery.append(" , " + str_today + "");

    // 쿼리를 데이터베이스에 전달하여 실행
    res = stmt.executeUpdate(cQuery.toString());

    // 쿼리 실행 중 오류 여부에 따른 반환값 설정
    if (res == 1) returnStr = "Y";
    else returnStr = "N";
}
return returnStr;

} catch (Exception e) {
    throw new Exception(e.getMessage());
}
}

// 오퍼레이션: 카테고리상품목록조회
// 파라미터: 카테고리명(String)
// 반환값: 카테고리에 속한 상품목록(Vector)
public Vector getItemList(String ctgr_name) throws Exception
{
    // 데이터베이스에 접근하고, 쿼리 수행 결과를 저장하기 위한 변수 선언
    DBConnectionPoolMgr dbMgr = null;
    Connection conn = null;
    Statement stmt = null;
    ResultSet query_result = null;

    // 반환값 변수 선언
    Vector return_vector = new Vector();

    try {
        // 데이터베이스에 접근하기 위한 라이브러리에서 사용할 변수 설정
        dbMgr = DBConnectionPoolMgr.getInstance();
        conn = dbMgr.getConnection();
        stmt = conn.createStatement();
    }

```



```

// 쿼리 문자열 생성
StringBuffer query = new StringBuffer();
query.append("SELECT item_name, cost, reg_date");
query.append("FROM t_item");
query.append("WHERE ctgr_name = " + ctgr_name + "");

// 쿼리를 데이터베이스에 전달하여 실행
query_result = stmt.executeQuery(query.toString());

// 쿼리 수행 결과를 반환값 변수에 저장
while (query_result.next()) {
    ItemDAO item_list = new ItemDAO();
    item_list.itemName = query_result.getString("item_name");
    item_list.cost = query_result.getInt("cost");
    item_list.regDate = query_result.getString("reg_date");
    return_vector.add(item_list);
}

return return_vector;

} catch (Exception e) {
    throw new Exception(e.getMessage());
}
}

// 오퍼레이션: 상품상세조회
// 파라미터: 상품명(String)
// 반환값: 상품정보(ItemDAO)
public ItemDAO getItemDetail(String item_name) throws Exception
{
    // 데이터베이스에 접근하고, 쿼리 수행 결과를 저장하기 위한 변수 선언
    DBConnectionPoolMgr dbMgr = null;
    Connection conn = null;
    Statement stmt = null;
    ResultSet query_result = null;

    // 반환값 변수 선언
    ItemDAO item_detail = new ItemDAO();

    try {
        // 데이터베이스에 접근하기 위한 라이브러리에서 사용할 변수 설정
        dbMgr = DBConnectionPoolMgr.getInstance();
        conn = dbMgr.getConnection();
        stmt = conn.createStatement();
    }
}

```

```

// 쿼리 문자열 생성
StringBuffer query = new StringBuffer();
query.append("SELECT item_name, description, item_img, cost, reg_date");
query.append("FROM t_item");
query.append("WHERE item_name = '" + item_name + "'");

// 쿼리를 데이터베이스에 전달하여 실행
query_result = stmt.executeQuery(query.toString());

// 쿼리 수행 결과를 반환값 변수에 저장
query_result.next();
item_detail.itemName = query_result.getString("item_name");
item_detail.description = query_result.getString("description");
item_detail.itemImg = query_result.getString("item_img");
item_detail.cost = query_result.getInt("cost");
item_detail.regDate = query_result.getString("reg_date");

return item_detail;
} catch (Exception e) {
    throw new Exception(e.getMessage());
}
}
}

```

데이터베이스 생성

DBMS에서 테이블을 생성하기 위한 명령어는 다음과 같다. 각자의 DBMS를 사용하여 데이터베이스를 생성하는 과정은 나타나지 않았다. 다음은 데이터베이스를 생성한 다음 설계 문서에서 정의된 테이블 t_category와 t_item을 생성하기 위한 명령어 스크립트이다.

```

CREATE TABLE t_category (
    ctgr_name VARCHAR(50) NOT NULL PRIMARY KEY,
    up_ctgr_name VARCHAR(50)
)
CREATE TABLE t_item (
    item_name VARCHAR(50) NOT NULL PRIMARY KEY,
    description VARCHAR(500) NOT NULL,
    item_image VARCHAR(200),
    cost INT(9) NOT NULL,
    reg_date VARCHAR(8) NOT NULL,
    ctgr_name VARCHAR(50) NOT NULL
)

```



A p p e n d i x

C

인터넷 쇼핑몰 테스트 문서 예제

1. 인터넷 쇼핑몰 인수 테스트 계획서
2. 인터넷 쇼핑몰 인수 테스트 설계서
3. 인터넷 쇼핑몰 인수 테스트 케이스 명세서
4. 인터넷 쇼핑몰 인수 테스트 절차서
5. 인터넷 쇼핑몰 인수 테스트 상황 기록
6. 인터넷 쇼핑몰 인수 테스트 사건 보고서
7. 인터넷 쇼핑몰 인수 테스트 요약 보고서

1 인터넷 쇼핑몰 인수 테스트 계획서

테스트 계획서 식별자

/*이 테스트 계획서의 고유 식별자를 명시한다.*/

문서 번호: 'TPlan_A'

서론

/*테스트할 소프트웨어 항목과 소프트웨어 특성을 요약한다. 각 항목의 필요성과 그 이력을 포함시킬 수 있다.

다음 문서들이 존재할 경우, 최상위 수준의 테스트 계획서에는 이들에 대한 참조를 포함하도록 한다.

- 프로젝트 승인서
- 프로젝트 계획서
- 품질 보증 계획서
- 형상 관리 계획서
- 관련 정책
- 관련 표준

여러 수준의 테스트 계획서에서, 각 하위 수준의 계획서는 바로 위 상위 수준의 계획서를 참조하여야만 한다.*/

(1) 목적

이 문서는 인터넷 쇼핑몰 소프트웨어의 인수 테스트 수행을 지원하기 위한 테스트 항목, 테스트 환경, 테스트 수행 결과 등을 정의하기 위한 목적으로 기술되었다.

(2) 범위

이 테스트 계획서는 인터넷 쇼핑몰의 기능 중 관리자가 사용하는 기능에 대한 인수 테스트

계획서를 범위로 한다.

(3) 관련 문서

RequirementSpec_A: 인터넷 쇼핑물 요구사항 명세서

테스트 항목

/*테스트 항목의 버전/개정 수준을 포함하여 이들을 식별한다. 또한 하드웨어 요구사항에 영향을 주는 전달 매체의 특성을 명시하거나, 테스트를 시작하기 전의 논리적 또는 물리적 변환의 필요성을 표시한다(예를 들면, 프로그램들은 테이프에서 디스크로 옮겨야만 한다).

다음의 테스트 항목 문서화에 대한 참조가 있다면 이를 제공한다.

- 요구사항 명세서
- 설계 명세서
- 사용자 지침서
- 운영 지침서
- 설치 안내서

테스트 항목과 관련된 사고(incident) 보고서는 모두 참조한다.

테스트에서 특별히 제외되어야 할 항목들을 식별할 수도 있다.*/

이 테스트에서는 분석 단계와 설계 단계의 산출물에 기술된 시스템의 기능이 실제로 완전하게 구현되었는지를 시험한다. 다음 유스케이스에서 기술한 기능을 대상으로 테스트를 실시한다.

- UC_A02 카테고리등록
- UC_A05 상품등록

테스트 대상 특성

/*테스트할 소프트웨어의 특성과 그 조합을 모두 식별하고, 이들 각각에 관련된 테스트 설계 명세서를 식별한다.*/

이 문서의 테스트 항목은 다음과 같은 특성을 갖는다.

- UC_A02 카테고리등록: 관리자용 UI를 통해 카테고리를 입력받으며, 정상 입력된 결과는 데이터베이스에 저장되고, 일반 사용자 UI의 검색 조건에 카테고리 항목이 추가된다.
- UC_A05 상품등록: 관리자용 UI를 통해 상품의 상세정보를 입력하며, 정상 입력된 결과는 데이터베이스에 저장되고, 일반 사용자 및 관리자 UI에서 입력된 상품의 목록 및 상세 결과를 확인할 수 있다.

테스트 대상이 아닌 특성

/[*테스트에서 제외할 모든 특성과 특성들의 중요한 조합, 그리고 이들을 제외시키는 사유를 식별한다.*]/

이 문서에서는 다음의 사항에 대해서는 고려하지 않는다.

- 시스템의 성능(반응 속도)은 시스템을 구동하는 PC 환경 또는 네트워크 상태에 따라 영향을 받을 수 있으므로 테스트에서 고려하지 않는다.
- 동시에 여러 사용자가 접속하여 서버에 부하가 걸리는 상황은 이 테스트에서 고려하지 않는다.

접근 방법

/[*테스트에 대한 전반적인 접근 방법을 기술한다. 특성이나 특성 조합의 중요한 그룹 각각에 대하여, 이들 특성 그룹이 적절히 테스트된다는 것을 보증하는 접근 방법을 명시한다. 지정한 특성 그룹들을 테스트하는 데 사용할 도구, 기법 및 주요 활동을 명시한다.

주요 테스트 작업을 식별할 수 있고, 이들 각각을 테스트하는 데 필요한 시간을 예측할 수 있도록 접근 방법을 충분하고 상세히 기술하도록 한다.

희망하는 최소한의 포괄성을 명시하고, 테스트 시도(effort)의 포괄성을 판단하는 데 사용할 기법을 식별한다(예를 들어, 어떤 명령문이 적어도 한 번 실행되었는지 확인하는 것). 그 외에 완료 기준(예를 들어, 에러 빈도)을 모두 명시한다. 요구사항을 추적하기 위한 기법도 명시하도록 한다.

테스트 항목의 가용성, 테스트 자원의 가용성 및 최종 기한과 같은 테스트에 대한 중요한 제약 사항을 식별한다.*]/

인터넷 쇼핑몰의 카테고리등록 및 상품등록 기능의 테스트를 위해 사용자(관리자)가 자주 유발할 수 있는 사항들을 중점적으로 테스트한다. 전체 데이터에 대한 정상 입력 상황, 필수 항목 미입력 상황, 잘못된 데이터 입력 상황 등을 종합적으로 고려하여 어떠한 상황에서도

정상 데이터가 입력됨을 확인할 수 있도록 테스트한다.

테스트 결과는 사용자 인터페이스와 데이터베이스의 내용을 기반으로 확인한다.

항목의 성공/실패 기준

/[*각 테스트 항목들이 테스트에 합격인지 불합격인지를 결정할 기준을 명시한다.*]/

- UC_A02 카테고리등록
 - 정상 데이터 입력의 경우 데이터베이스와 사용자 UI에 정상적으로 등록된 카테고리 가 나타날 경우 성공
 - 비정상 데이터 입력의 경우 오류 메시지 출력 후 재입력할 경우 성공
 - 비정상 데이터가 데이터베이스에 입력될 경우 실패
- UC_A05 상품등록
 - 정상 데이터 입력의 경우 데이터베이스와 사용자 UI에 정상적으로 등록된 상품이 나타날 경우 성공
 - 비정상 데이터 입력의 경우 오류 메시지 출력 후 재입력할 경우 성공
 - 비정상 데이터가 데이터베이스에 입력될 경우 실패

일시 중지 기준 및 재개 요구사항

/[*이 계획서와 관련된 테스트 항목에 대한 테스트 활동의 일부 또는 전부를 일시 중지시키는 데 사용할 기준을 명시하고, 테스트가 재개되었을 경우 반복해야 하는 테스트 활동을 명시한다.*]/

테스트 수행 중 실패의 경우가 발생할 때 모든 테스트는 중지하고, 오류를 수정한 이후 재개한다. 이때 잘못된 코드를 변경할 경우 기존 테스트에 대해서 모두 새롭게 테스트하여 변경된 코드가 시스템에 영향을 미치지 않음을 확인해야 한다.

테스트 인도물

테스트 종료 후 다음과 같은 산출물을 인도한다.

- 테스트 계획서
- 테스트 설계 명세서

- 테스트 케이스 명세서 및 테스트 데이터
- 테스트 절차 명세서
- 테스트 상황 기록
- 테스트 사고(incident) 보고서
- 테스트 요약 보고서

테스트 입력 데이터 및 테스트 출력 데이터를 인도물로서 식별하도록 한다. 테스트 도구 (예를 들어, 모듈 드라이버와 스텐브)도 포함될 수 있다.

테스트 작업

/ *테스트를 준비하고 수행하는 데 필요한 일련의 작업을 식별한다. 모든 작업 간의 내부적 의존성을 식별하며, 필요한 기술은 어떤 것이든 모두 식별한다.*/

해당 사항 없음

환경 요건

/ *필요하거나 희망하는 테스트 환경의 속성을 명시한다. 여기에는 하드웨어를 포함한 설비의 물리적 특성, 통신 및 시스템 소프트웨어, 사용 모드(예를 들어, 단독형) 그리고 테스트를 지원하는 데 필요한 그 외의 소프트웨어나 공급품을 모두 포함하도록 한다. 또한 테스트 설비, 시스템 소프트웨어, 소프트웨어, 데이터 및 하드웨어와 같은 자산 등을 위하여 대비해야만 하는 보안 수준도 명시한다.

필요한 특정 테스트 도구를 식별한다. 그 외에 테스트에 필요한 것들(예를 들어, 간행물이나 사무실 공간)을 모두 식별한다. 현재 테스트 그룹에게 제공되지 않은 모든 필수품에 대하여 이를 확보할 수 있는 출처를 식별한다.*/

(1) 하드웨어 환경

인터넷 쇼핑물의 실제 운용 환경을 고려하여 서버/클라이언트 환경에서 테스트한다.

(2) 소프트웨어 환경

인터넷 쇼핑물 환경의 특성상 일반적으로 사용하는 Windows/Linux/iOS에서 정상 동작해야 하며, Internet Explorer/Chrome/Firefox/Safari에서 정상 동작해야 한다.

책임

/[*관리, 설계, 준비, 실행, 입회, 대조 및 해결 등을 담당할 그룹을 식별한다. 또한 식별한 테스트 항목들과 환경 요건을 준비할 담당 그룹을 식별한다.

이들 담당 그룹들은 개발자, 테스트 수행자, 운영 요원, 사용자 대표, 기술 지원 요원, 데이터 관리 요원, 품질 지원 요원 등을 포함할 수 있다.*/

인터넷 쇼핑물 인수 테스트와 관련하여 각 조직은 다음과 같은 책임을 갖는다.

(1) 인수 테스트 수행 팀

인터넷 쇼핑물 인수 테스트를 총괄하여, 테스트에 대한 전문 지식을 제공한다.

(2) 개발 팀

테스트 수행 간 발생한 오류를 수정하고, 데이터베이스 등 시험 결과를 기록하는 데 협조한다.

구성원 및 교육 요건

/[*테스트 요원이 갖추어야 할 요건을 기술 수준별로 명시한다. 필요한 기술을 제공하기 위한 교육과정을 식별하여 선택할 수 있도록 한다. */

테스트 수행을 위한 기본적인 시스템 사용 방법과 요구사항 명세서의 내용에 대한 기본적인 교육을 시험 시작 2일 전에 실시한다.

일정

/[*모든 항목 전달 사건(item transmittal events)뿐만 아니라, 소프트웨어 프로젝트 일정에서 식별된 테스트 이정표(milestone)를 포함한다.

그 외 필요한 테스트 이정표를 모두 정의하고, 각 테스트 작업을 수행하는 데 필요한 예상 시간을 산출한다. 각 테스트 작업과 테스트 이정표에 대한 일정을 명시하며, 각 테스트 자원(설비, 도구 및 인원)에 대하여 그 활용 기간을 명시한다.*/

인터넷 쇼핑물의 인수 테스트는 20××년 8월 31일까지 완료한다.

위험 요소 및 비상 대처 상황

/*테스트 계획서에서 위험도가 높을 것으로 예상되는 요소들을 식별하고, 이들에 대한 비상 대처 계획을 명시한다(예를 들면, 테스트 항목의 전달이 늦어질 경우, 인도일을 맞추기 위해서는 추가 공수 투입 등).*/

테스트 수행 중 심각한 오류가 발생하여 시스템 수정에 많은 시간이 추가될 경우 개발 팀에 중견 개발자 1명을 추가 배치하여야 한다.

승인

/*계획서를 반드시 승인하여야 하는 모든 사람들의 직책과 이름을 명시한다. 서명과 날짜를 기록할 공간을 마련한다.*/

테스트 관리자

날짜

개발 프로젝트 관리자

날짜

품질 보증 관리자

날짜

2 인터넷 쇼핑몰 인수 테스트 설계서

테스트 설계 명세서 식별자

/*이 테스트 설계 명세서의 고유 식별자를 명시한다. 관련된 테스트 계획서에 대한 참고가 있다면 이를 제공한다.*/

문서 번호: TSpec_A

이 문서는 인터넷 쇼핑몰 인수 테스트 항목 중 카테고리등록 기능에 대한 테스트 설계서이다.

테스트 대상 특성

/*테스트 항목을 식별하고, 이 설계 명세서의 대상인 특성과 그 조합을 기술한다. 그 외의 특성들은 테스트해 볼 수 있으나, 식별할 필요는 없다.

각 특성 또는 특성 조합에 대하여, 항목 요구사항 명세서나 설계 기술서에 있는 이들과 관련된 요구사항의 참조를 포함하도록 한다.*/*

인터넷 쇼핑물의 카테고리등록 기능은 다음과 같은 입력 데이터 특성을 갖는다.

- 최상위카테고리: 기존에 등록되어 있는 카테고리 임의의 문자열(length: 20)
- 하위카테고리: 기존에 등록되어 있는 카테고리 또는 새롭게 입력할 카테고리 임의의 문자열(length: 20)

카테고리의 조합 특성은 카테고리 자체의 임의의 문자열에 대한 조합과 카테고리의 계층적 구조의 조합 특성을 갖는다.

- 카테고리 문자열 조합: 임의의 모든 문자의 조합
- 카테고리 계층 조합: 상위카테고리를 지정할 경우 하위카테고리, 상위카테고리를 지정하지 않은 경우 최상위카테고리

세부 접근 방법

/*테스트 계획서에 기술된 방법들을 상세하게 명시한다. 사용되는 특정 테스트 기법들을 포함한다. 테스트 결과를 분석하는 방법을 식별하도록 한다(예를 들어, 비교 프로그램이나 육안 검사).

테스트 케이스의 선정에 대한 이론적 근거를 제공하는 분석은 모두 그 결과를 명시한다. 예를 들어, 여러 허용 범위를 결정할 수 있는 조건을 명시할 수 있다(예를 들면, 부적절한 입력과 적절한 입력을 구분하는 조건).

어떤 테스트 케이스에나 공통인 속성을 요약한다. 여기에는 관련된 일련의 테스트 케이스의 모든 입력에 반드시 적용되는 입력 제약 사항과, 공유된 환경 요건, 공유된 특정 절차 요구사항, 공유된 테스트 케이스의 의존성은 어떠한 것이든 모두 포함할 수 있다.*/*

사용자(관리자)로부터 입력받을 수 있는 카테고리 문자열 중 정상 시나리오와 예외 시나리오에 대하여 먼저 테스트를 수행한 후 조합 특성을 반영한 데이터에 대하여 테스트를 수행한

다. 개별 카테고리의 입력 가능 범위인 length 20을 기준으로 경계값 분석 기법을 적용한다. 또한 카테고리의 계층을 확인할 수 있도록 최상위카테고리 입력과 하위카테고리 입력을 확인할 수 있도록 테스트한다.

테스트 식별

/ *테스트 설계와 관련된 테스트 케이스의 식별자를 열거하고, 이들 각각에 대하여 간략히 설명한다. 어떤 특별한 테스트 케이스가 하나 이상의 테스트 설계 명세서에서 식별될 수 있다. 이 테스트 설계 명세서와 관련된 절차의 식별자를 열거하고, 이들 각각에 대하여 간략히 설명한다. */

테스트 케이스 식별자	설명	유효/무효
IS.CAT.TS001	length(카테고리 문자열)=0	무효
IS.CAT.TS002	0 <length(카테고리 문자열) <=20	유효
IS.CAT.TS003	length(카테고리 문자열) > 20	무효
IS.CAT.TS004	상위카테고리 선택 ×	유효
IS.CAT.TS005	상위카테고리 선택 ○	유효

특성의 성공/실패 기준

/ *특성 또는 특성의 조합이 합격인지 불합격인지를 결정하는 데 사용할 기준을 명시한다. */

(1) 카테고리 문자열 길이 기준(length: 20) 특성의 성공/실패 기준

- 정상적인 입력 범위인 0 <length(카테고리 문자열) < 20이 입력될 경우 카테고리 조합의 경우에 따라 최상위카테고리 또는 하위카테고리에 정상적으로 등록된 경우 성공, 그 이외의 경우 실패
- 예외적인 상황인 length(카테고리 문자열)=0, length(카테고리 문자열) > 20에 대하여 정상 등록된 경우 실패, 재입력 유도 메시지 출력 후 재입력 화면으로 전환되는 경우 성공

(2) 카테고리 계층 구조 특성의 성공/실패 기준

- 상위카테고리를 선택하지 않고 카테고리를 입력할 경우 최상위카테고리 목록에 입력 카테고리가 나타날 경우 성공, 하위카테고리로 등록되거나 나타나지 않을 경우 실패

- 상위카테고리를 선택하고 카테고리를 입력할 경우 선택한 상위카테고리의 하위 목록에 입력 카테고리가 나타날 경우 성공, 선택한 카테고리의 하위에 나타나지 않거나 등록되지 않은 경우 실패

③ 인터넷 쇼핑몰 인수 테스트 케이스 명세서

테스트 케이스 명세서 식별자

/*이 테스트 케이스 명세서의 고유 식별자를 명시한다.*/

문서 번호: TCase_A

이 문서는 인터넷 쇼핑몰 인수 테스트 항목 중 카테고리등록 기능에 대한 테스트 케이스 명세서이다.

테스트 항목

/*테스트 케이스에 의해 테스트를 수행할 항목과 특성을 식별하고, 이에 대하여 간략하게 기술한다.

각 항목마다, 다음 테스트 항목 문서들에 대한 참조 제공을 고려한다.

- 요구사항 명세서
- 설계 명세서
- 사용자 지침서
- 운영 지침서
- 설치 안내서

*/

인터넷 쇼핑몰의 카테고리등록 기능에 대한 테스트는 테스트 설계서 TSpec_A에 명시된 테스트 케이스에 대하여 실시하며, 카테고리 문자열 길이 특성과 카테고리 계층 구조 특성을 반영한 테스트 케이스에 대하여 테스트를 수행한다.

테스트 케이스 식별자	설명	유효/무효
IS.CAT.TS001	length(카테고리 문자열)=0	무효
IS.CAT.TS002	0 < length(카테고리 문자열) <=20	유효
IS.CAT.TS003	length(카테고리 문자열) > 20	무효
IS.CAT.TS004	상위 카테고리 선택 x	유효
IS.CAT.TS005	상위 카테고리 선택 ○	유효

입력 명세서

/*테스트 케이스를 실행하기 위해 필요한 각 입력을 명시한다. 입력의 일부는 수치로(적절한 곳에 허용 오차도 함께) 명시할 수 있고, 상수 테이블이나 트랜잭션 파일 같은 것들은 이름으로 명시할 수 있다. 모든 적절한 데이터베이스, 파일, 터미널 메시지, 메모리 상주 영역, 그리고 운영 시스템에 의해 전달되는 값들을 식별한다.

입력들 간에 요구되는 모든 관계(예를 들어, 타이밍)를 명시한다.*/

카테고리등록 기능에는 두 개의 입력값을 갖는다. 각 테스트 케이스별로 독립적인 테스트 수행을 위하여 해당 특성에 대한 테스트를 수행할 때 다른 입력값은 정상인 상황을 고려한다.

- 상위카테고리항목(top_category)
- 카테고리이름(category_name)

각 특성에 대해서는 동등 분할 기법 및 경계값 분석 방법을 적용하여, 해당 특성의 경계값을 식별하여 입력 데이터를 선정하였다.

테스트 케이스 식별자	입력 데이터	상세 설명
IS.CAT.TS001	top_category : “의류”(selected) category_name : “”	0 < length <=20의 비정상 동작 범위의 최솟값인 length=0 적용
IS.CAT.TS002	top_category : “의류”(selected) category_name : “!”	0 < length <=20의 정상 동작 범위의 최솟값인 length=1 적용
	top_category : “의류”(selected) category_name : “기능성 활동성 여성 의류”	0 < length <=20의 정상 동작 범위의 최댓값인 length=20 적용
IS.CAT.TS003	top_category : “의류”(selected) category_name : “기능성 활동성 여성 의류!”	0 < length <=20의 비정상 범위의 최솟값인 length=21 적용
IS.CAT.TS004	top_category : “”(not selected) category_name : “아동 의류”	상위 카테고리 적용 동등 분할의 미선택 적용
IS.CAT.TS005	top_category : “의류”(selected) category_name : “남성 의류”	상위 카테고리 적용 동등 분할의 선택 적용

출력 명세서

/ *테스트 항목에 요구되는 출력과 특성(예를 들어, 응답 시간)을 모두 명시한다. 요구되는 출력이나 특성의 각각에 대하여 이들의 정확한 값(적절한 곳에 허용 오차와 함께)을 제공한다. */

테스트 케이스 식별자	예상 출력	예외 사항
IS.CAT.TS001	“카테고리이름을 입력하세요” 오류 메시지 출력	정상 등록되거나, 오류 메시지 출력되지 않음
IS.CAT.TS002	“의류” 카테고리의 하위카테고리에 “!” 정상 출력	의류 카테고리의 하위카테고리에 “!”가 없거나, 기타 메시지 출력
	“의류” 카테고리의 하위카테고리에 “기능성 활동성 여성 의류” 정상 출력	의류 카테고리의 하위카테고리에 “기능성 활동성 여성 의류”가 없거나, 기타 메시지 출력
IS.CAT.TS003	“카테고리이름은 20자를 넘길 수 없습니다.” 오류 메시지 출력	정상 등록되거나, 오류 메시지 출력되지 않음
IS.CAT.TS004	최상위카테고리에 “아동 의류” 카테고리 출력	정상 등록되지 않거나, 최상위카테고리가 아님
IS.CAT.TS005	“의류” 카테고리의 하위카테고리로 “남성 의류” 카테고리 출력	정상 등록되지 않거나, “의류” 카테고리 하위카테고리에 “남성 의류”가 정상 출력되지 않음

환경 요건

(1) 하드웨어

/ *테스트 케이스를 실행하는 데 필요한 하드웨어의 특성과 구성도를 명시한다(예를 들면, 132자 * 24줄 CRT). */

해당 사항 없음

(2) 소프트웨어

/ *테스트 케이스를 실행하는 데 필요한 시스템과 응용 소프트웨어를 명시한다. 여기에는 운영체제, 시스템 소프트웨어, 컴파일러, 시뮬레이터 및 테스트 도구를 포함한다. */

테스트 케이스는 Windows/ Linux/ iOS 환경 및 Internet Explorer/Chrome/Firefox/Safari에서 모두 정상 동작임을 확인해야 한다.

(3) 기타

/[*독특한 설비의 요구나 특별히 훈련된 사람 등과 같은 기타 요구사항을 모두 명시한다.*]/

해당 사항 없음

특정한 절차 요구사항

/[*테스트 케이스를 실시하는 테스트 절차에 대한 특정한 제약 사항을 모두 기술한다. 이러한 제약 사항이란 특정한 준비 작업, 운영자 개입, 출력 확인 절차, 특정한 마감 작업 등이 될 수 있다.*]/

테스트 케이스 간의 의존성

/[*이 테스트 케이스 전에 반드시 실시되어야 하는 테스트 케이스들의 식별자를 나열한다. 그 의존성의 성격도 요약한다.*]/

해당 사항 없음

4 인터넷 쇼핑몰 인수 테스트 절차서

테스트 절차 명세서 식별자

/[*이 테스트 절차 명세서의 고유 식별자를 명시한다. 관련된 테스트 설계 명세서에 대한 참조를 제공한다.*]/

문서 번호: TProc_A

이 문서는 인터넷 쇼핑몰 인수 테스트 항목 중 카테고리등록 기능에 대한 테스트 절차서이다.

목적

/[*절차의 목적을 기술한다. 절차가 어떤 테스트 케이스를 실행하는 것이라면 그들의 각각

에 대한 참조를 제공한다.*/

이 문서는 인터넷 쇼핑몰의 카테고리등록 기능이 정상적으로 동작하는지 확인하기 위한 절차를 기술하는 데 목적이 있다.

특별 요구사항

/*절차의 실행에 필요한 특정 요구사항을 모두 식별한다. 여기에는 선행되어야 할 절차, 특정 기술 요건, 그리고 특정한 환경 요구사항이 포함된다.*/

해당 사항 없음

테스트 절차 단계

/*다음의 (1)~(10)까지의 단계들(steps)을 적용 가능하도록 포함시킨다.*/

(1) 상황 기록

/*테스트 실행 결과, 관측된 사고(incidents) 및 기타 테스트와 관련된 사건들(events)을 상황 기록에 기입하기 위한 특정한 방법이나 양식을 모두 기술한다.*/

인터넷 쇼핑몰의 인수 테스트에 대한 상황 기록은 테스트 상황 기록 보고서 양식에 따라 기술하며, 실패 및 예외 사항이 발생할 경우 테스트 사건 보고서 양식에 따라 작성한다. 또한 테스트 완료 후 테스트 결과 보고서를 통해 결과를 보고한다.

(2) 준비

/*절차를 실행하기 위하여 준비해야 할 조치 사항의 순서를 기술한다.*/

테스트 수행 전 'TSpec_A'에 기술된 테스트 케이스에 맞는 데이터를 선정한다.

(3) 시작

/*절차를 착수하는 데 필요한 조치 사항을 기술한다.*/

- ① 테스트 수행 전 인터넷 쇼핑몰 시스템을 구동한다.
- ② 관리자 권한으로 지정된 사용자로 로그인한다.
- ③ 카테고리등록 화면으로 이동한다.

(4) 진행

/[*절차 실행 중에 필요한 조치 사항을 모두 기술한다.*]/

- ① 최상위카테고리목록이 UI에 나타나는지 확인한다.
- ② 조회하고자 하는 카테고리를 선택한다.
- ③ 선택된 카테고리의 하위카테고리가 나타나는지 확인한다.
- ④ 상위카테고리를 선택한다.
- ④-1 최상위카테고리를 등록하고자 할 경우 선택하지 않는다.
- ⑤ 준비 단계에서 선정한 카테고리 문자열을 입력한다.
- ⑥ 저장 버튼을 누른다.

(5) 측정

/[*테스트 측정치를 산출할 방법을 기술한다(예를 들어, 네트워크 시뮬레이터를 사용하여 원격 터미널의 응답 시간을 측정하는 방법을 기술한다).*/]

- ① 유효 테스트 케이스에 대해 사용자 화면에 등록된 카테고리가 정상적으로 출력되는지 확인하고, 정상적으로 등록되지 않은 경우 테스트 사건 보고서에 기록한다.
- ② 무효 테스트 케이스에 대해 오류 메시지가 정상 출력되는지 확인하고, 무효 케이스가 등록되거나 오류 메시지가 출력되지 않은 경우 테스트 사건 보고서에 기록한다.

(6) 중단

/[*일정에 없는 사건(events)을 지시받아 테스트를 일시 중지하는 데 필요한 조치 사항을 기술한다.*]/

정상적으로 동작하지 않을 경우 테스트를 중단하고 문제가 발생한 부분을 수정한다.

(7) 재시작

/*절차상의 재시작점을 모두 식별하고, 재시작점 각각에 대하여 그 절차를 다시 수행하는 데 필요한 조치 사항을 기술한다.*/

오류 발생 부분을 수정한 이후 테스트를 재시작한다.

(8) 중지

/*실행을 지시에 의하여 중지하는 데 필요한 조치 사항을 기술한다.*/

오류 발생 후 수정하는 데 걸리는 시간이 24시간을 초과할 경우 테스트를 중지한다.

(9) 마감

/*환경을 복원하기 위한 필요한 조치 사항을 기술한다.*/

테스트를 위해 등록한 카테고리를 모두 삭제한 후 테스트를 종료한다.

(10) 비상 대처 상황

/*실행 중 일어날 수 있는 비정상적 사건(events)을 처리하는 데 필요한 조치 사항을 기술한다.*/

중지에 해당하는 오류가 발생할 경우, 테스트 관리자, 개발 프로젝트 관리자, 품질 보증 관리자의 회의를 통해 문제점을 식별한 후 오류 수정 기간, 재시작 일정 등을 포함한 대처 계획 보고서를 작성한다.

5 인터넷 쇼핑몰 인수 테스트 상황 기록**테스트 상황 기록 식별자**

/*이 테스트 상황 기록의 고유 식별자를 명시한다.*/

문서 번호: TEvent_A

이 문서는 인터넷 쇼핑몰 인수 테스트 항목 중 카테고리등록 기능에 대한 테스트 수행 상황 보고서이다.

설명

/※상황 기록의 등록 항목에서 특별하게 언급된 것을 제외하고 상황 기록의 모든 등록 항목에 적용되는 정보를 여기에 포함하도록 한다. 다음 사항들을 고려하여 기술하도록 한다.

- ① 테스트할 항목을 이들의 버전/개정 정도를 포함하여 식별한다. 각 항목에 대해 전달 보고서에 대한 참조가 있으면 이를 제공한다.
- ② 테스트가 실시되는 환경의 속성을 식별한다. 설비 명칭, 사용할 하드웨어(예를 들어, 사용할 메모리 양, CPU 모델 번호, 테이프 드라이브의 개수와 모델, 대량 저장 장치), 사용되는 시스템 소프트웨어, 이용 가능한 메모리의 양과 같은 가용 자원 등을 여기에 포함한다.*/

인터넷 쇼핑몰의 1차 인수 테스트이며, 20××년 8월 10일 실 관리자(홍길동)에 의하여 실시되었다.

활동 및 사건(event) 기입

/※각 사건에 대해 활동 시작과 종료를 포함하여, 발생 날짜 및 시간을 기록자의 이름, 직책 등과 함께 기록한다.*/

20××-08-10	사건
14:00 홍길동, 테스트 시작	
14:15 IS.CAT.TS001~3에 대한 테스트 실시	
14:30 유효 문자열 중 특수 문자 비정상 동작 확인	AP01-01
사고 보고서를 작성	
15:00 홍길동, 테스트 관리자, 개발 프로젝트 관리자, 품질 보증 관리자의 회의를 통해 1차 테스트 중단 결정	

/※다음 사항을 고려하여 추가 정보를 기술할 수 있다. */

(1) 실행 설명

/*실행할 테스트 절차의 식별자와 그 명세서에 대한 참조를 기록한다. 테스트 수행자, 운영자, 참관자를 포함하여 테스트 절차 실행에 참석한 모든 사람들을 기록하며, 이들의 역할도 함께 기록한다.*/

(2) 절차의 결과

/*각 실행에 대하여 육안으로 관측할 수 있는 결과를 기록한다(예를 들어, 발생된 에러 메시지, 시스템의 중지, 운영자 조치 요구사항 등). 또한 모든 출력물의 위치를 기록하고(예를 들어, 릴 번호), 테스트 실행의 성공 또는 실패 여부를 기록한다.*/

(3) 환경 정보

/*등록 항목에 특수한 환경 조건을 모두 기록한다(예를 들어, 하드웨어의 대체).*/

(4) 예외적 사건(event)

/*예기치 못했던 사건이 생기면 그 전후에 발생한 내용을 기록한다(예를 들어, 요약 화면을 요청하여 화면이 올바르게 나왔으나 응답 시간이 비정상적으로 길다. 반복 시도해 봐도 응답 시간이 동일하게 길게 나온다). 테스트 절차를 시작할 수 없게 하거나 완료할 수 없게 만드는 상황이 있으면 이를 기록한다(예를 들어, 정전 또는 시스템 소프트웨어 문제).*/

정상 입력 범위를 갖는 카테고리 문자열에 특수 문자가 입력될 경우 사용자 화면에 정상 출력되지 않은 상황이 발생함

해당 문자열 = <?

(5) 사고(incident) 보고서 식별자

/*테스트 사고 보고서가 발행될 때마다 이들의 식별자를 기록한다.*/

테스트 수행과정에서 발생한 AP01-01은 사건 보고서 AP01-01을 통해 기술한다.

6 인터넷 쇼핑몰 인수 테스트 사건 보고서

테스트 사고 보고서 식별자

/[*이 테스트 사고 보고서의 고유 식별자를 명시한다.*]/

문서 번호: AP01-01

요약

/[*사고를 요약한다. 여기에 관련된 테스트 항목을 이들의 버전/개정 수준을 표시하여 식별한다. 적절한 테스트 절차 명세서, 테스트 케이스 명세서 및 테스트 상황 기록에 대한 참조를 제공하도록 한다.*]/

AP01-01은 1차 인수 테스트 과정에서 TProc_A에 대해 IS.CAT.TS002의 테스트 수행과정에서 발생한 사건으로 자세한 상황은 TEvent_A에 기술하였다.

사고 설명

/[*사고에 대해 설명한다. 여기에는 다음 항목들을 포함하도록 한다.

- 입력
- 예상 결과
- 실제 결과
- 예외 사항
- 날짜와 시간
- 절차의 단계(procedure step)
- 환경
- 반복 시도 횟수
- 테스트 수행자
- 참관자

사고 원인을 분리시켜 정정하는 데 도움을 줄 수 있는 관련 활동과 관측 사항을 포함하도록 한다. 예를 들어, 그 사고와 연관이 있을 수 있는 테스트 케이스의 실행을 모두 기술하고,

발행된 테스트 절차로부터 벗어난 사항을 모두 기술한다.*/

20XX-08-10 홍길동

카테고리등록 기능의 테스트 과정에서 IS.CAT.TS002에 대한 데이터로 특수 문자열 또는 키워드를 입력할 경우 정상 동작하지 않음을 확인하였다. 해당 테스트에서 입력한 데이터는 “<?”로 HTML상에 javascript 구문의 시작을 나타내는 문자열이다. 이러한 특수 문자열을 입력하더라도 입력한 문자열이 그대로 출력되는 것을 기대하였으나, javascript 오류와 함께 해당 카테고리 이후의 내용이 정상 출력되지 않음을 확인하였다.

영향

/사고가 테스트 계획서, 테스트 설계 명세서, 테스트 절차 명세서 또는 테스트 케이스 명세서에 미치는 영향에 대하여 알려진 것이 있다면 이를 표시한다.*/

이 사건으로 특별히 영향받는 요소는 없으나, 해결될 때까지 테스트 활동을 중단한다.

7 인터넷 쇼핑몰 인수 테스트 요약 보고서

테스트 요약 보고서 식별자

/이 테스트 요약 보고서의 고유 식별자를 명시한다.*/

문서 번호: TResult_A

이 문서는 인터넷 쇼핑몰 인수 테스트 결과 요약 보고서이다.

요약

/테스트 항목의 평가를 요약한다. 테스트한 항목을 버전/개정 수준을 표시하여 식별한다. 테스트 활동이 발생한 환경을 표시한다.

각 테스트 항목마다 다음 문서에 대한 참조를, 이들 문서가 있다면 제공한다: 테스트 계획서, 테스트 설계 명세서, 테스트 절차 명세서, 테스트 항목 전달 보고서, 테스트 상황 기록,

테스트 사고 보고서*/

인터넷 쇼핑몰의 인수 테스트 항목에 대해 총 5개의 결함을 수정한 후 모든 테스트를 통과하였다. 해당 테스트와 관련하여 다음과 같은 문서를 참조한다.

TPlan_A 인터넷 쇼핑몰 인수 테스트 계획서
 TSpec_A 인터넷 쇼핑몰 인수 테스트 설계서
 TProc_A 인터넷 쇼핑몰 인수 테스트 절차서
 TEvent_A 인터넷 쇼핑몰 인수 테스트 상황 기록
 AP01-01 인터넷 쇼핑몰 인수 테스트 사건 보고서

변동 사항

/*설계 명세서로부터 벗어난 테스트 항목의 변동 사항을 모두 기록한다. 테스트 계획서, 테스트 설계서, 테스트 절차서로부터 벗어난 변동 사항을 모두 표시하고, 각 변동 사항에 대한 사유를 명시한다.*/

테스트 중에 식별된 조건들은 원래 기능 설계에서 기술한 부적절한 조건들을 강화시켰으며, 이것은 11개의 테스트 케이스 명세서를 추가하도록 하였다. 이러한 변경 사항은 현재의 문서에 모두 반영되었다.

포괄성 심사

/*테스트 계획서가 존재할 경우, 계획서에 명시된 포괄성 기준에 따라 테스트 프로세스의 포괄성을 평가한다. 충분히 테스트되지 않은 특성이나 특성의 조합을 식별하고, 그 이유를 설명한다.*/

테스트 계획서에 명시된 기능과 테스트 설계서에 정의된 테스트 케이스를 기반으로 모든 테스트를 통과하였다.

결과 요약

/*테스트 결과를 요약한다. 모든 해결된 사고를 식별하고, 그 해결방법을 요약한다. 해결되지 않은 사고도 역시 모두 식별한다.*/

3개의 테스트 케이스(IS.CAT.TS002, IS.REG.TS010, IS.BUY.TS004)에 의해 예외적인 상황 처리에 대한 결함을 확인하였으며, 이를 해결하기 위해 별도의 로직을 추가하였고, 몇 개의 새로운 테스트 케이스를 정의하였다. 기존 테스트 케이스와 새롭게 추가된 테스트 케이스를 기반으로 재테스트를 수행하였다. 모든 특성이 이들 테스트를 통과하였다.

평가

/*각 테스트 항목에 대한 전반적인 평가를 이들의 제한 사항을 포함하여 기록한다. 이러한 평가는 테스트 결과와 항목의 성공/실패 기준에 의하여 이루어져야 한다. 실패 시 예상되는 위험 사항을 포함할 수 있다.*/*

인터넷 쇼핑몰 시스템의 인수 테스트 과정에서 단지 3개의 주요 결함을 발견하였으며 모든 결함이 수정되어, 현재 식별된 오류는 없다.

활동 요약

/*주요 테스트 활동과 사건(event)을 요약한다. 투입한 자원에 대한 데이터(예를 들어, 주요 테스트 활동 각각에 대하여 활용한 전체 요원들의 수준, 총 기계 사용 시간, 총 소요 시간 등)를 요약한다.*/*

테스트 시작 : 20XX-08-01	예측 시간	실제 시간
테스트 설계(케이스 포함)	2일	3일
테스트 수행	2일	2일
오류 수정	2일	1.5일
테스트 보고	0.5일	0.5일
테스트 종료 : 20XX-08-08	6.5일	7일

승인

/*이 보고서를 반드시 승인해야 하는 모든 사람들의 이름과 직책을 명시한다. 서명과 날짜를 기록할 공간을 마련한다.*/*

개발 프로젝트 관리자

날짜



A p p e n d i x

D

인터넷 쇼핑몰 프로젝트 현장

프로젝트 현장 개요

프로젝트 헌장(project charter) 개요

프로젝트명	인터넷 쇼핑몰 개발 프로젝트		
작성자	김철수	문서 버전	Rev. 1.0
프로젝트 목적 및 사유	• 인터넷 쇼핑몰을 구축하여 오프라인뿐만 아니라 온라인에서도 영업을 할 수 있도록 지원		
프로젝트 성공 기준 및 결정권자	• 프로젝트의 성공 기준: 주어진 예산 규모로 프로젝트 수행 기간인 6개월(22주) 내에 쇼핑몰을 개발 완료하여 고객에게 인도하여 활용 • 프로젝트 성공 결정권자: 쇼핑몰 시스템을 인도받은 고객이 결정		
프로젝트 범위 및 요구사항	• 프로젝트 범위: 인터넷 쇼핑몰의 구축 및 쇼핑몰을 운영할 수 있는 환경 조성 • 프로젝트 요구사항: Internet Explorer뿐만 아니라 다른 웹 브라우저에서도 사용할 수 있는 시스템 구축		
프로젝트 위험 요소	• 프로젝트에 필요한 숙련된 인력 부족(Java, JSP, 객체지향 프로그래밍) • 프로젝트의 일정이 연기되면 계획했던 인터넷 쇼핑몰 운영이 어려워 금전적인 손실의 발생이 불가 피하다. 이를 방지하기 위해 철저한 일정 관리가 필요하다. • 시스템의 불량 또는 결함에 의해 쇼핑몰 기업 이미지에 손해를 끼치는 일이 없도록 철저한 품질 관리가 이루어져야 한다.		
프로젝트 요구 인력	• 프로젝트 관리자(1명): 프로젝트의 전반적인 관리 및 책임 • 프로젝트 품질 관리 인원(1명): 프로젝트 개발 일정 중 정해진 시기에 품질 관리 활동 수행 • 프로젝트 요구사항 분석가(1명): 시스템 요구사항 분석 수행 • 프로젝트 개발 팀(3명): 시스템 설계, 개발 및 테스트 단계에서 업무 수행 • 프로젝트 테스터(1명): 개발이 완료된 후 확인 시험 단계에서 요구사항 분석가가 테스트 업무 수행 • 프로젝트 감사 팀(외부 업체): 정해진 시기에 감사를 실시하고, 고객의 요구에 따라 추가적으로 감사를 실시할 수 있다.		
Milestone	Milestone		목표 종료 날짜
	프로젝트 시작일		2013-07-01
	요구사항 분석 완료		2013-08-04
	개발 완료		2013-11-03
	시스템 인도 및 프로젝트 종료		2013-12-01
프로젝트 예산 규모	예산 : 총 72,760,000원		

프로젝트 관리자 책임 및 권한	책임	<ul style="list-style-type: none"> 프로젝트 세부 일정 계획 및 수행: 관리자는 프로젝트 소요 자원, 세부 일정을 계획하여 프로젝트 관리 계획서에 기록하고, 이를 바탕으로 프로젝트를 수행해야 한다. 프로젝트 관리 계획서, 요구사항 명세서, 설계 문서에 대하여 고객, 프로젝트 관리자가 서명하고 공식화한다. 위의 산출물이 공식화된 후 수정이 요구되는 경우 수정의 원인을 밝히고, 수정의 원인이 프로젝트 팀에 있을 경우 프로젝트 팀에서 책임을 부담한다. 프로젝트를 수행하기 전에 프로젝트 관리자는 프로젝트의 위험 요소를 미리 파악하고 대책을 세워두어야 한다. 위험이 발생했을 때, 발생한 위험이 사전에 파악되지 않은 위험일 경우 관리자는 고객의 도움을 받지 못하고 스스로 해결해야 한다.
	권한	<ul style="list-style-type: none"> 프로젝트 구성원 선발 및 교체 권한: 관리자는 프로젝트 인원 선발 및 교체 권한을 갖는다. 프로젝트 예산에 대한 권한: 관리자는 허용된 예산 범위 내에서 프로젝트의 예산 계획을 작성한다. 이 예산 계획은 추후 프로젝트 관리 계획서에 포함해야 한다. 프로젝트 예산 사용 승인 권한: 프로젝트의 예산은 관리자의 승인을 받아 집행한다. 예산 사용 내역은 별도로 기록해야 하며, 감사 팀이나 고객의 요구가 있다면 언제라도 제출해야 한다.
고객의 권한 및 책임	책임	<ul style="list-style-type: none"> 고객은 프로젝트의 요구사항을 프로젝트 팀에 제공해야 한다. 고객은 공식기술검토회의에 참여하여 프로젝트 관리 계획서, 요구사항 명세서, 설계 문서에 대한 검토와 피드백을 수행한다. 프로젝트 수행 시 작성되는 산출물이 공식화된 후 고객의 요구사항에 변경이 발생할 경우, 변경에 필요한 비용은 고객이 부담한다. 변경 비용은 프로젝트 관리자와 협의하여 결정한다. 고객은 프로젝트의 품질 관리 및 형상 관리 업무에 참가한다.
	권한	<ul style="list-style-type: none"> 소프트웨어를 포함한 프로젝트의 산출물은 고객이 소유권을 갖는다. 고객은 프로젝트의 지연이 발생할 경우, 지연의 정도에 따라 프로젝트 관리자에게 페널티를 부과할 수 있다. <ul style="list-style-type: none"> 0~1주 미만: 예산의 0.5% 1~1주: 예산의 1% 1주 초과: (초과한 주)×1%
종료 승인 요건	<ul style="list-style-type: none"> 감사 팀의 검수를 통과한다. 고객의 인수 시험을 통과한다. 고객에게 소프트웨어를 전달하고, 소프트웨어 설치를 수행한다. 계약 시 납품하기로 한 산출물들을 고객에게 전달한다. 	
승인	프로젝트 관리자: 김철수 (인)	고객 대표: 최명수 (인)



Appendix

E

인터넷 쇼핑몰 프로젝트 관리 계획서

1. 개요 프로젝트 요약
2. 참고 문헌
3. 핵심 용어 정의 및 약어 용어 | 약어
4. 프로젝트 조직 및 책임 외부 조직 | 내부 조직 | 역할과 책임
5. 관리 프로세스 계획 초기 계획 | 작업 계획 | 통제 계획 | 위험 관리 계획 | 종료 계획
6. 기술 프로세스 계획 프로세스 모형 | 방법, 도구와 기법 | 기반 구조 계획 | 제품 수락 계획
7. 지원 프로세스 계획 형상 관리 계획 | 검증과 확인 계획 | 문서화 계획 | 품질 보증 계획 | 검토 계획

문서 번호 : SEIshop-SPMP

인터넷 쇼핑몰 프로젝트

프로젝트 관리 계획서(PMP)
(Project Management Plan)

Rev. 1.0

Approval :

_____	____/____/____
_____	____/____/____
_____	____/____/____

Date

개정 현황					
V1.00	2013. 7. 14	최초 작성	박영수		
개정 번호	일자	변경 사유	작성	검토	승인

1 개요

프로젝트 요약

(1) 목적 및 범위

이 프로젝트는 고객에게 온라인 쇼핑몰 시스템을 구축·제공하여 오프라인뿐만 아니라 온라인에서도 영업이 가능하도록 지원하는 것을 목적으로 한다. 이 프로젝트에서 개발하고자 하는 인터넷 쇼핑몰 시스템의 개발 범위는 다음과 같다.

- 물품을 구매하려는 소비자는 인터넷 쇼핑몰에 접속하여 물품의 상세 정보를 확인할 수 있으며 물품 구매를 할 수 있다.
- 관리자는 카테고리의 분류체계를 관리하고, 상품을 특정 카테고리에 포함시켜 등록한다.
- 상품을 구매하기 위해서는 먼저 원하는 상품을 장바구니에 담고, 장바구니에 담긴 상품들 중에서 선택적으로 주문할 수 있다.
- 상품 구매 시 결제 수단은 온라인 입금과 신용카드 결제로 구분한다. 온라인 입금은 बैं킹 시스템을 통해 입금 여부를 확인한 후 결제 처리하며, 신용카드 결제는 신용카드 인

중 회사로의 결제 승인 서비스를 통해 즉시 처리가 가능하도록 한다.

- 쇼핑몰은 인터넷 익스플로러(Internet Explorer)뿐만 아니라 다양한 웹 브라우저에서도 작동 가능하도록 개발한다.

(2) 가정 사항과 제약 사항

해당 사항 없음

(3) 프로젝트 인도물

이 프로젝트 완료 시 고객에게 납품되어야 할 인도물은 다음과 같다.

- 프로젝트 관리 계획서
- 설계 문서
- 품질 보증 계획서
- 소스 코드
- 형상 관리 계획서
- 시험 계획서
- 검증 및 확인 계획서
- 시험 결과서
- 요구사항 명세서
- 사용자 매뉴얼

(4) 일정과 예산 요약

① 일정 : 2013년 7월 1일~2013년 12월 1일(6개월, 총 22주)

작업	일정											
	7월	8월	9월	10월	11월	12월						
A. 계획												
프로젝트 관리 계획 수립												
B. 분석												
요구사항 정의												
프로토타이핑												
분석 품질 보증 활동												
C. 설계												
기본 설계												
상세 설계												
설계 품질 보증 활동												
D. 구현												
코딩												
구현 품질 보증 활동												

E. 시험												
시험 계획												
시험 수행												
시험 품질 보증 활동												
F. 설치 및 점검												
설치												
점검												

② 예산: 총 72,760,000원

- 인건비: 54,000,000원
프로젝트 관리자(PM)(1명)
프로젝트 요구사항 분석가(1명)
개발자(3명)
테스터(1명)
- 제비용: 18,760,000원
Server급 컴퓨터(1대) : 쿼드코어 CPU, 8GB RAM, 3TB HDD 이상

② 참고 문헌

한국정보통신기술협회 TTAS.KO-11.0024, 소프트웨어 프로젝트 관리 계획서 표준
IEEE Std 1058-1998, Standard for Software Project Management
PMI, Project Management Body of Knowledge

③ 핵심 용어 정의 및 약어

용어

- 프로젝트(project): 유일한 제품, 서비스 또는 결과를 창출하기 위해 수행하는 한시적인 활동
- 이해관계자(stakeholder): 프로젝트에 적극적으로 참여하거나 프로젝트의 결과에 긍정적 또는 부정적 영향을 미치는 개인 또는 조직

- 작업 분할 구조(WBS): 프로젝트 범위(scope)를 분할하여 구조화한 것
- 작업 패키지(WP): WBS의 최하위 수준 요소로 일의 가장 작은 단위

약어

- PM: Project Manager, 프로젝트 관리자
- WP: Work Package, 작업 패키지
- WBS: Work Breakdown Structure, 작업 분할 구조

4 프로젝트 조직 및 책임

프로젝트 수행 중 개발에 직접 참여하는 내부 조직과 개발에 직접 참여하지는 않지만 프로젝트 수행에 중요한 외부 조직으로 프로젝트 조직을 구분할 수 있다.

외부 조직

외부 조직으로는 개발 대상인 쇼핑몰 시스템을 최종 인수할 고객과 프로젝트 수행 중 정기적인 감사 업무를 수행한 감사 팀, 프로젝트 산출물이 고객의 요구사항 범위를 모두 만족하였는지를 확인하는 품질 보증 팀으로 구성한다.

내부 조직

프로젝트 수행을 위해 프로젝트 관리자(PM)는 프로젝트 팀을 구성한다. 프로젝트 팀의 구성원으로는 요구사항 분석가, 개발 팀, 테스트 팀으로 구성하며 통합 테스트 단계에서는 독립된 테스트 팀원을 통해 시험을 수행하도록 한다.

역할과 책임

인터넷 쇼핑몰 프로젝트에 참여하는 조직 및 담당자의 역할과 책임을 정의한다.

구분		책임
외부	고객	<ul style="list-style-type: none"> 요구사항 제시 최종 인수 테스트 참여
	외부 감사 팀	<ul style="list-style-type: none"> 정기/비정기 감사 실시
	품질 보증 팀	<ul style="list-style-type: none"> 품질 보증 업무 수행
내부	프로젝트 관리자	<ul style="list-style-type: none"> 프로젝트 개발 총괄 책임
	요구사항 분석가	<ul style="list-style-type: none"> 고객의 요구사항 분석 개발 쇼핑몰 테스트 수행
	개발 팀	<ul style="list-style-type: none"> 자료 수집 쇼핑몰 시스템 설계 프로그램 개발 및 테스트, 디버깅 개발 문서 및 매뉴얼 작성 주요 회의 참석
	테스트 팀	<ul style="list-style-type: none"> 개발 쇼핑몰 테스트 수행

5 관리 프로세스 계획(Managerial Process Plans)

초기 계획

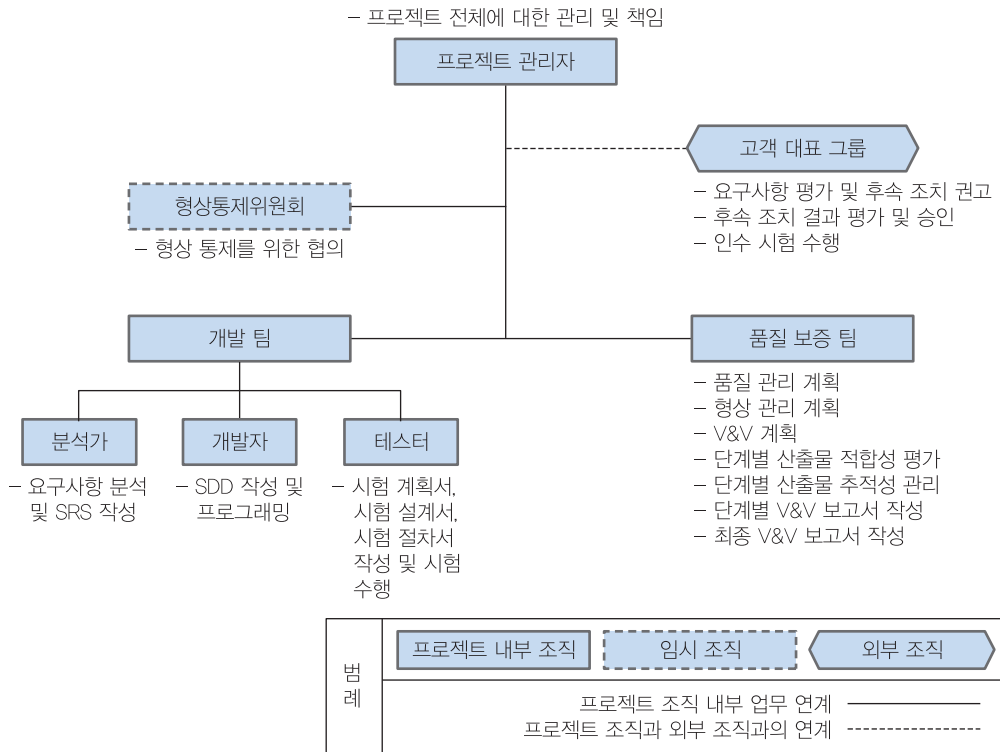
프로젝트 초기 계획(set-up plan)은 프로젝트 착수를 위한 초기 비용 및 일정 산정 계획, 팀원 계획, 자원 획득 계획, 훈련 계획 등을 말한다. 프로젝트 규모와 범위에 따라 이러한 계획들은 다른 계획에 편입되어 운용될 수 있다.

(1) 산정 계획

프로젝트의 비용, 일정, 자원은 프로젝트가 진행되는 동안 주기적으로 산정해야 한다.

(2) 팀원 계획

이 프로젝트를 수행하는 동안 필요한 조직 및 인원 구성은 그림 1과 같다.



::그림 1 | 프로젝트 조직 구성도

(3) 자원 획득 계획

프로젝트 완수를 위해 필요한 자원은 다음과 같다.

개발 팀/테스트 인력에 대한 획득 계획: 내부 인력 개발 × 팀 인력 3인 100% 투입

• 개발 환경

구분	개발자 환경	쇼핑몰 서비스 운영 환경
운영체제	WINDOWS XP	UNIX
데이터베이스	MYSQL SERVER 4.1	MYSQL SERVER 4.1
웹 개발 언어	PHP4, HTML	PHP4, HTML
하드웨어	아파치 서버	아파치 서버

- 교육: 프로그래밍 교육 2회(온라인 교육 포함)
- 설비: 사무실, 전화, 전기 설비, 책상, 인터넷 망

앞에 명시한 자원 중 사무실 확보가 제약 사항이 될 수 있으나 이는 실험실 활용을 통해 해결할 수 있다.

(4) 프로젝트 팀원 훈련 계획

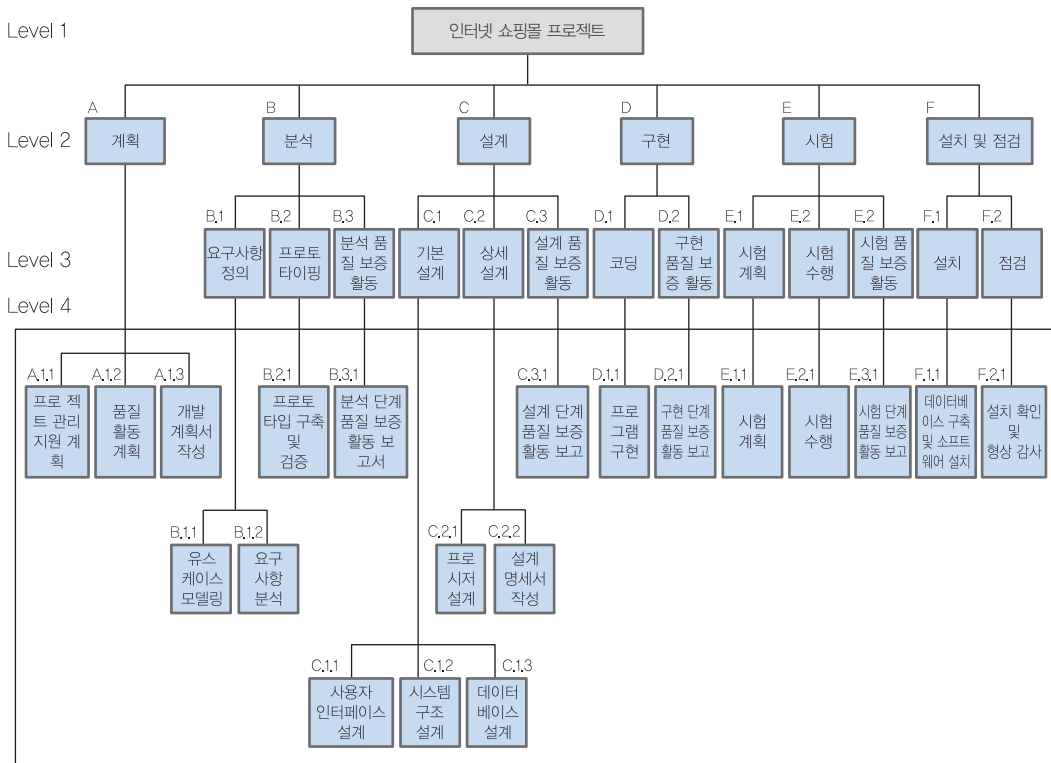
프로젝트 수행에 필요한 기술 수준 확보를 위해 필요한 훈련은 다음과 같다.

- 훈련 유형: Java, JSP, 객체지향 프로그램 언어 습득
- 훈련받을 인원: 개발 팀원
- 훈련 방법: 온라인 프로그램 개발 강좌 수강

작업 계획

(1) 작업 활동

프로젝트에서 수행해야 하는 작업 활동(work activities)은 아래의 WBS(Work Breakdown Structure, 작업 분할 구조)로 나타낼 수 있다. WBS를 통해 작업 패키지(WP)를 파악하고 각 활동들의 관계를 파악하여 필요한 기간과 자원 요구사항을 산정한다.



::그림 2 | 인터넷 쇼핑몰 시스템 WBS

(2) 일정 할당

① Critical Path Method(CPM) - 작업 리스트

작업	선행 작업	소요 기간(주)
A. 계획		
A.1.1 프로젝트 관리 지원 계획		2
A.1.2 품질 활동 계획		1
A.1.3 개발 계획서 작성	A.1.1, A.1.2	1
B. 분석		
B.1.1 유스케이스 모델링		1
B.1.2. 요구사항 분석(객체지향 분석)	B.2.1	3
B.2.1 프로토타입 구축 및 검증	B.1.1	1
B.3.1 분석 단계 품질 보증 활동 보고	A.1.3, B.1.2	1
C. 설계		
C.1.1 사용자 인터페이스 설계	B.1.1	2
C.1.2 시스템 구조 설계	B.1.2	1
C.1.3 데이터베이스 설계	B.1.2	1
C.2.1 프로시저 설계	C.1.2	3
C.2.2 설계 명세서 작성	C.1.1, C.1.3, C.2.1	1
C.3.1 설계 단계 품질 보증 활동 보고	B.3.1, C.2.2	1
D. 구현		
D.1.1 프로그램 구현	C.2.2	8
D.2.1 구현 단계 품질 보증 활동 보고	C.3.1, D.1.1	1
E. 시험		
E.1.1 시험 계획(단위, 통합, 시스템, 인수)	C.2.2	2
E.2.1 시험 수행(단위, 통합, 시스템, 인수)	D.1.1, E.1.1	2
E.3.1 시험 단계 품질 보증 활동 보고	D.2.1, E.1.1, E.2.1	1
F. 설치 및 점검		
F.1.1 데이터베이스 구축 및 소프트웨어 설치	E.2.1	1
F.2.1 설치 확인 및 형상 감사	E.3.1, F.1.1	1

② 프로젝트 상세 일정

프로젝트 수행 단계	일정
계획	2013-07-01~2013-07-21
분석	2013-07-01~2013-08-04
설계	2013-08-05~2013-09-08
구현	2013-09-09~2013-11-03
시험	2013-11-04~2013-11-24
설치 및 점검	2103-11-25~2013-12-01

(3) 자원 할당

① 인력: 팀원-7명

② 인건비: PM 5,000,000원 × 6개월 = 30,000,000원

팀원 2,000,000원 × 6개월 × 6명 = 24,000,000원

③ 제비용

가. 국내 여비(대전~서울)

110,000원/인 × 3회 × 7명 = 2,310,000원

나. 인쇄비

사용자 매뉴얼: 18,000원 × 5권 = 90,000원

개발 문서: 10,000원 × 5권 × 3set = 150,000원

다. 참고 문헌비: 30,000원 × 10권 = 300,000원

라. 재료비

레이저 프린터 토너: 150,000원 × 10개 = 1,500,000원

개발 환경 구축용 PC: 2,500,000원 × 5대 = 12,500,000원

A4 용지: 250/매 × 3,000원 × 50권 = 150,000원

마. 회의비: 30,000원/인 × 6회 × 10명 = 1,800,000원

제비용 합계: 18,760,000원

④ 총비용 = 인건비 + 제비용 = 72,760,000원

(4) 예산 할당

각 WP별 원가를 산정하여 프로젝트 수행을 위한 활동에 필요한 예산을 할당한다.

통제 계획

통제는 프로젝트의 작업 수행이 제대로 이루어지는지 감시하는 활동과 밀접한 관련이 있다. 통제는 계획 대비 실제 프로젝트 수행 성과를 비교한 뒤 시정 및 예방 조치를 취하고 대응 방안을 실행한다. 일정에 대한 기준은 일정 기준선이며 예산에 대한 기준은 원가 기준선이다. 계획에 따라 프로젝트가 완료될 수 있도록 프로젝트 기간 동안 통제 업무를 수행한다.

(1) 요구 통제 계획

범위 기준선에 대한 변경을 관리한다. 고객의 요구사항이 통제가 안 된 상태에서 승인 없이 조금씩 늘어나는 것을 방지하기 위하여 범위 기준선을 확정하고 이를 지속적으로 감시한다. 요구사항 명세서, 프로젝트 관리 계획서, 요구사항 추적 매트릭스는 범위 기준선이 되며 계획과 실적에 대한 차이 정보를 이해 당사자에게 보고한다. 이때 인도물의 진행 정도, 완료된 인도물에 대한 실적 정보를 포함하여 제시한다.

(2) 일정 통제 계획

일정 통제의 기준인 일정 기준선에 따라 성과 검토를 수행하고 차이를 분석하여 필요 시 변경 요청/처리한다. 진행 중인 활동은 남은 기간, 완료율, 시작한 날짜, 종료 예정일 등의 일정 성과를 측정하여 비교, 분석한다.

일정 성과는 획득 가치(Earned value)를 통해 측정할 수 있다.

- 자원 평준화: 일정의 차이를 맞추기 위해 필요 시 자원(인력, 예산 등) 투입을 증가시킬 수 있다.
- 가정 시나리오 분석: 일정 차이로 인한 영향을 분석하고 일정 계획을 맞추기 위한 분석을 수행한다.
- 일정 선도 및 일정 단축: 필요 시 자원을 추가 투자하여 일정을 단축한다.
- 일정 계획 도구 사용: MS 프로젝트 프로그램을 사용하여 일정을 계획하고 관리한다.

일정 변경이 일어날 경우 변경 요청 사항들을 관련된 문서에 반영하여 프로젝트가 완료될 수 있도록 조치한다.

(3) 예산 통제 계획

계획된 예산 내에서 프로젝트를 완수하기 위해 예산 할당을 통해 원가 기준선을 수립한다.

이에 대해 변경을 관리하고 예산을 갱신하기 위해 프로젝트 수행 기간 동안 지속적으로 예산 사용을 감시한다.

주요 예산 통제 활동은 다음과 같다.

- 원가 기준선에 변경을 유발하는 요소들을 확인하고 관련된 조치를 취함
- 모든 변경 요청이 적시에 실행되도록 지속적으로 감시
- 원가 초과가 발생할 경우 승인된 예산 총액을 넘지 않도록 유지
- 발생한 원가가 계획에 명시된 원가 기준선을 지키는지 주기적으로 파악
- 변경에 대한 기록을 관리하고 승인된 변경 사항은 이해 당사자에게 통보
- 원가가 초과될 경우 수용 가능한 한계 내로 유지되도록 시정 조치 수행

승인된 비용 대비 실제 집행된 비용에 대한 정보를 수집하여 작업 성과 정보(Work performance information)를 프로젝트 관리자 및 이해관계자에게 주기적으로 보고한다. 성과 측정 및 보고를 위해 획득 가치 관리(EVM: Earned Value Management) 기법을 사용한다. EVM을 통해 예산 성과가 좋은지 나쁜지 판단하며 최종 사업비를 예측하여 필요 시 변경 요청을 통해 프로젝트 예산 관리를 수행한다.

(4) 품질 통제 계획

품질 통제는 프로젝트 수행 결과가 품질 표준에 부합하는지를 결정하기 위해 결과를 감시하고 결과가 품질 표준에 부합하지 않으면 왜 그런지 이유를 찾아 제거하는 것을 포함한다. 프로젝트의 산출물인 쇼핑물 시스템은 인도물이 되며 품질 통제 시 검토되어야 할 대상이다. 품질 통제는 ‘인터넷 쇼핑물 품질 보증 계획서’에 따라 수행한다.

(5) 보고 계획

프로젝트를 수행하면서 생기는 정보들은 매월 성과 보고서로 작성하여 관련 이해관계자들에게 배포한다.

- 성과 보고서: 매월 일정 성과, 비용 성과 정보 및 향후 예측 정보 포함
- 배포 방법: 월간회의 보고 또는 이메일 배포

위험 관리 계획

쇼핑물 개발 프로젝트의 위험 요소를 식별하고 위험도를 분석하여 대책을 수립하고 프로

젝트 수행 기간 동안 지속적으로 위험 요소를 관리한다.

::표 1 | 위험 식별 목록

No	위험 목록	위험도	대책
1	숙련된 인력 부족 (Java, JSP, 객체지향 프로그래밍)	상	<ul style="list-style-type: none"> • 온라인 교육 실시 • 개발 경험자 신규 채용
2	짧은 프로젝트 일정	중	<ul style="list-style-type: none"> • 철저한 일정 관리 • 일정 관리 도구 및 추가 근무를 통한 일정 단축
3	개발 시스템의 품질 확보	중	<ul style="list-style-type: none"> • 품질 관리를 통한 고객 요구사항 만족
4	고객과의 의사소통	중	<ul style="list-style-type: none"> • 산출물의 문서화 • 성과 보고서 작성 및 월간 회의 수행

고객이 원하는 온라인 쇼핑물의 형상을 파악하고자 프로토타입 기법을 사용하고, 각 개발 단계의 산출물을 문서화하여 고객과의 의사소통 문제를 해결하고자 한다. 또한 고객의 요구가 분석 단계 이후에도 계속해서 변경될 우려가 있으므로 요구사항 명세서를 만들어 충분히 검토하고 변경 요구를 통제/동결하도록 한다.

종료 계획

프로젝트 종료 시 고객은 프로젝트 수행 결과 생성된 산출물과 인도물인 쇼핑물 시스템을 인도받는다. 이때 프로젝트 관리 계획과 요구사항 명세서에 명시된 고객의 요구사항은 모두 충족되어야 한다.

앞에서 제시한 인도물들은 프로젝트 종료 시 고객에게 전달되어야 하며, 고객은 납품 확인서를 발행하여 공식적으로 프로젝트를 종료시킨다. 또한 프로젝트 팀은 프로젝트를 수행하는 동안 생성된 계약서, 개발 문서 등을 프로젝트 팀 자산으로 등재하여 관리한다.

6 기술 프로세스 계획

프로세스 모형

기본적으로 폭포수 모델을 바탕으로 소프트웨어 개발이 이루어지며, 고객의 요구사항을 프로젝트 초기에 규명하기 위해 프로토타입을 구축하여 확인한다.

작업	소요 기간(주)	완료해야 할 프로젝트 인도물
A. 계획		프로젝트 관리 계획서
A.1.1 프로젝트 관리 지원 계획	2	품질 보증 계획서
A.1.2 품질 활동 계획	1	형상 관리 계획서
A.1.3 개발 계획서 작성	1	검증 및 확인 계획서
B. 분석		
B.1.1 유스케이스 모델링	1	
B.1.2. 요구사항 분석(객체지향 분석)	3	요구사항 명세서
B.2.1 프로토타입 구축 및 검증	1	
B.3.1 분석 단계 품질 보증 활동 보고	1	
C. 설계		
C.1.1 사용자 인터페이스 설계	2	
C.1.2 시스템 구조 설계	1	
C.1.3 데이터베이스 설계	1	설계 명세서
C.2.1 프로시저 설계	3	
C.2.2 설계 명세서 작성	1	
C.3.1 설계 단계 품질 보증 활동 보고	1	
D. 구현		
D.1.1 프로그램 구현	8	소스 코드
D.2.1 구현 단계 품질 보증 활동 보고	1	
E. 시험		
E.1.1 시험 계획(단위, 통합, 시스템, 인수)	2	시험 계획서
E.2.1 시험 수행(단위, 통합, 시스템, 인수)	2	시험 결과서
E.3.1 시험 단계 품질 보증 활동 보고	1	
F. 설치 및 점검		
F.1.1 데이터베이스 구축 및 소프트웨어 설치	1	사용자 매뉴얼
F.2.1 설치 확인 및 형상 감사	1	

방법, 도구와 기법

온라인 쇼핑몰 전 개발과정에 객체지향 개발방법론을 활용하고 요구사항 분석 및 설계 단계에 유스케이스와 UML을 이용한 문서화가 이루어진다. 각 단계가 끝나면 품질 보증 활동을 통해 고객과 개발자가 서로 개발 사항을 확인하면서 개발 업무를 진행하여 고객의 요구에 대해 최대한 충족시켜 줄 수 있도록 노력한다.

기반 구조 계획

프로젝트 개발 환경 및 완성된 온라인 쇼핑몰이 운영될 환경은 다음과 같다.

구분	개발자 환경	쇼핑몰 서비스 운영 환경
운영체제	WINDOWS XP	UNIX
데이터베이스	MYSQL SERVER 4.1	MYSQL SERVER 4.1
웹 개발 언어	PHP4, HTML	PHP4, HTML
하드웨어	아파치 서버	아파치 서버

또한 프로젝트 수행 시 다음과 같은 표준을 준수하도록 한다.

- IEEE Std 1058-1998 IEEE Standard for Software Project Management Plan
- IEEE Std 730-1998 IEEE Standard for Software Quality Assurance Plans
- IEEE Std 828-1998 IEEE Standard for Software Configuration Management Plans
- IEEE Std 1012-1986 IEEE Standard for Software Verification and Validation Plans

제품 수락 계획

쇼핑몰의 개발이 완료되면 정해진 절차에 따라 고객에게 제품을 인도한다. 제품을 인도하는 절차는 다음과 같다.

개발 완료	<ul style="list-style-type: none"> • 소프트웨어가 완성되면 인수 시험을 실시하여 통과하였을 경우에 제품을 인도하도록 한다. • 인수 시험의 통과 여부는 고객과 프로젝트 관리자가 사전에 협의한 시험 계획서(인수 시험)에 나타나 있는 기준에 따라 결정한다.
납품 및 설치	<ul style="list-style-type: none"> • 계약 시 납품하기로 한 산출물들을 고객에게 전달한다. • 개발 팀은 고객에게 소프트웨어를 전달하고, 소프트웨어 설치를 수행한다. • 설치된 소프트웨어에 이상이 없을 경우 확인서를 작성하고 인도 절차를 마무리한다.

7 자원 프로세스 계획

형상 관리 계획

인터넷 쇼핑몰 시스템 개발 과정 및 운영 유지보수 과정에서 변화하는 소프트웨어의 모습을 가시화하고 그 변경 내용을 체계적이며 일관성 있게 수용하도록 하여 시스템의 품질을 보증한다. 이를 위해 형상 관리 계획을 수립하고 프로젝트 개발 기간 동안 적용한다. 형상 관리란 형상 식별, 형상 통제, 형상 보고, 형상 감사 등을 포함하여 관리하며 별도로 작성한 ‘온라인 쇼핑몰 형상 관리 계획서’에 따라 관리한다.

검증과 확인 계획

프로젝트 수행을 통해 개발 완료한 쇼핑몰 시스템에 대하여 고객에게 인도되기 전에 고객의 요구사항을 충족하였는지 확인하고 검증 절차를 수행한다. 이때 검증 대상인 쇼핑몰 시스템에 대한 검증 기준으로 프로젝트 관리 계획서와 요구사항 명세서를 이용하여 검사(inspection)를 통해 확인·검증한다.

문서화 계획

프로젝트 수행 중 개발 관련 자료는 체계적으로 기술하여 문서화(documentation)한다. 작성된 문서는 동료 검토를 수행하고 PM의 승인을 받는다. 반드시 승인된 개발 문서를 기준으로 형상 관리 기준선을 확정하고 이를 형상 관리와 품질 관리에 활용한다. 또한 승인된 문서는 이해관계자에게 배포하여 개발 시 참고하도록 한다.

품질 보증 계획

인터넷 쇼핑몰 시스템의 품질을 보증하기 위해 품질 계획, 품질 지원 및 심사, 품질 표준 등 품질 관리에 대한 체계를 정립하고, 활동별로 요구되는 품질 기준을 제시한다. 이 프로젝트에서는 ‘온라인 쇼핑몰 품질 보증 계획서’에 따라 고객의 요구사항이 만족될 수 있도록 품질을 관리한다. 품질 보증 계획서에서는 품질 보증 조직 및 역할을 제시하고 프로젝트가 수행되는 동안 관리되어야 할 대상을 식별하며, 품질 보증 활동에 대해 상세히 작성하여 관리한다.

검토 계획

온라인 쇼핑몰 구축 프로젝트를 진행하면서 각 산출물에 대한 주요 검토 회의의 일정과 참석자는 다음과 같다.

검토 회의	일시	참석 예정자
요구사항 검토 회의	8/4	고객, 프로젝트 팀
상세 설계 검토 회의	9/8	고객, 프로젝트 팀, 테스트 팀
시제품 구현 검토 회의	11/3	고객, 프로젝트 팀, 테스트 팀
인수	12/1	고객, 프로젝트 팀

검토 회의에서는 각 단계별 개발 진행 사항을 검토하고 다음 단계로의 진행 여부를 결정한다. 검토 결과 부족한 사안에 대해서는 조치 항목(action item)으로 선정하고 완료 목표일을 설정하여 다음 검토 회의에서 다시 검토하여 고객의 요구사항이 충족될 수 있도록 노력한다.



Appendix

F

인터넷 쇼핑몰 품질 보증 계획서

1. 목적
2. 범위
3. 핵심 용어 정의 및 약어 용어 | 약어
4. 참고 문헌
5. 품질 보증 조직 및 역할 품질 조직 | 책임 | 활동 영역 | 역할 및 책임
6. 산출물 목적 | 최소 요구 산출물 목록
7. 표준, 실행, 지침, 매트릭스 목적 | 소프트웨어 품질 관리 방안
8. 소프트웨어 품질 보증 활동 소프트웨어 시험 | 외부 감리 | 수정 활동 | 공동 기술, 관리 검토
9. 주요 일정 계획 사업 추진 일정 | 품질 보증 세부 일정 계획

문서 번호: SEIshop-SQAP

인터넷 쇼핑몰 프로젝트

소프트웨어 품질 보증 계획서(SQMP) (Software Quality Assurance Plan)

Rev. 1.0

Approval :

_____	_____/_____/_____
_____	_____/_____/_____
_____	_____/_____/_____

Date

개정 현황					
V1.00	2013. 7. 12	최초 작성	박영수		
개정 번호	일자	변경 사유	작성	검토	승인

1 목적

인터넷 쇼핑물 시스템의 품질을 보증하고자 품질 계획, 품질 지원 및 심사, 품질 표준 등 품질 관리에 대한 체계를 정립하고, 활동별로 요구되는 품질 기준을 제시하는 데 목적이 있다.

2 범위

이 문서는 인터넷 쇼핑물 프로젝트의 품질 목표 달성을 위해 프로젝트 수행 조직의 품질 보증 활동에 대하여 품질 관리 업무 지원 및 품질 심사의 기준이 된다. 인터넷 쇼핑물 프로젝트의 품질 관리는 착수부터 종료까지 전 사업 기간에 걸쳐서 이루어진다.

3 핵심 용어 정의 및 약어

용어

- 품질: (1) 기본 특성이 요구사항을 충족하는 정도(PMBOK), (2) 그것의 유용성을 결정하는 성질 또는 그것의 사용 또는 적용 목적을 만족(또는 충족)시켰는가 그러지 못했는가를 결정하기 위한 평가의 대상이 되는 고유의 성질이나 성능(소프트웨어 프로젝트 관리), (3) 요구사항에 대한 적합 또는 사용성에 대한 적합(프로젝트 관리의 이해)
- 소프트웨어 품질: (1) 제품을 생산하고 사용하는 모든 사람들에게 측정 가능한 가치를 주는 유용한 제품을 생산하는 데 효과적으로 적용할 수 있는 소프트웨어 프로세스 (pressman), (2) 주어진 요구사항 또는 사용 목적을 만족할 수 있는 능력을 가지고 있는 소프트웨어 제품의 특성과 특징의 총체(소프트웨어 프로젝트 관리), (3) 주어진 요구사항을 만족시킬 수 있는 소프트웨어 능력의 총량(IEEE)
- 품질 비용: 품질을 추구하거나 품질 관련 행동을 수행하는 데 있어 발생하는 모든 비용과 품질 결여로 발생하는 모든 비용. 품질 비용은 예방 비용(prevention cost), 평가 비용(appraisal cost), 장애 비용(failure cost)을 포함
- 품질 관리: 품질 요구사항을 충족시키는 데 사용되는 운영상의 기법 및 활동
- 품질 보증: 품질 요구사항을 충족시킨다는 신뢰감을 주기 위하여 품질 시스템 내에서 실시되는 모든 계획적이고 체계적인 활동

약어

- QA: Quality Assurance, 품질 보증

4 참고 문헌

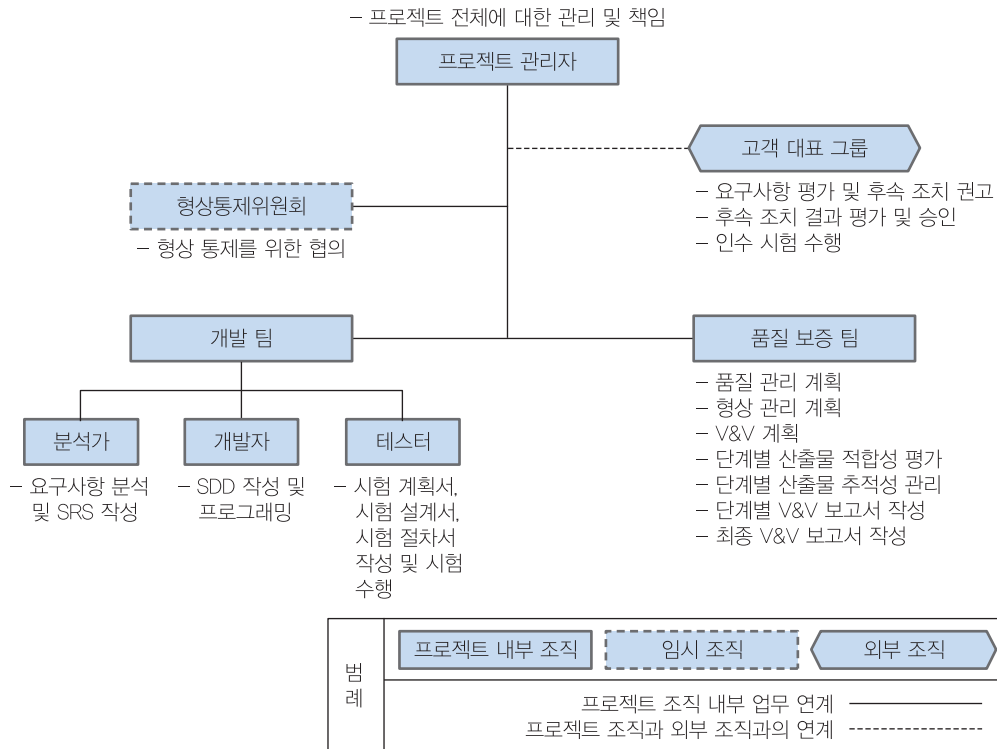
- TTAS, JE-730.1, “소프트웨어 품질 보증 계획 수립을 위한 지침”, 한국정보통신기술협회, 2003.
- IEEE Std 703-1998, “IEEE Standard for Software Quality Assurance Plan”, IEEE, 1998.
- 고석하, 홍정유, 『소프트웨어 프로젝트 관리』, 생능출판사, 2007.
- Ian Sommerville 저, 권기태 역, 『소프트웨어 공학』, 홍릉과학출판사, 2008.
- Roger S. Presman 저, 김성규, 김성철, 이숙희 역, 『소프트웨어 공학』, 한산, 2011.
- 김병철, 『프로젝트 관리의 이해』, 세화, 2003.

안재성, 『쉽게 합격하는 PMP』, JSFactory, 2008.

정기원, 윤창섭, 김태현, 『소프트웨어 프로세스와 품질』, 홍릉과학출판사, 1997.

5 품질 보증 조직 및 역할

품질 조직



::그림 1 | 프로젝트 조직 구성도

책임

::표 1 | 역할과 책임

수행 조직	역할 및 책임
품질 보증 팀	<ul style="list-style-type: none"> • 품질 보증 계획 수립 • 품질 요구사항 조사 • 품질 보증 교육 실시 • 계획/표준/절차 지원 및 검토 • 프로세스 점검 및 평가 • 산출물 점검 및 평가 • 품질 점검 보고서 작성 • 품질 평가 보고서 작성 • 품질 보증 결과 자료 수집 및 분석
프로젝트 관리자	<ul style="list-style-type: none"> • 품질 보증 계획서 검토 • 품질 점검 요청 • 품질 점검 보고서 검토 • 품질 평가 요청 • 품질 평가 보고서 검토
개발 팀	<ul style="list-style-type: none"> • 요구 분석, 설계, 프로그래밍 등의 개발 활동 수행 • 개발 단계별 산출물 작성 • 품질 점검 및 평가에 따른 시정 조치 • 품질 표준 문서, 품질 관리 기준 적용

활동 영역

(1) 품질 관리 대상 정의

::표 2 | 품질 관리 대상

개발 단계	품질 관리 대상 항목	식별자	검토 시점	중점 검토 사항
계획	프로젝트 관리 계획서	SPMP	계획 단계 완료 후	<ul style="list-style-type: none"> • 계획 타당성 • 계획의 일관성
	품질 보증 계획서	QAP	계획 단계 완료 후	<ul style="list-style-type: none"> • 계획 타당성 • 계획의 일관성
	형상 관리 계획서	SCMP	계획 단계 완료 후	<ul style="list-style-type: none"> • 계획 타당성 • 계획의 일관성
	검증 및 확인 계획서	SVWP	계획 단계 완료 후	<ul style="list-style-type: none"> • 계획 타당성 • 계획의 일관성
분석	요구사항 명세서	SRS	분석 단계 완료 후	<ul style="list-style-type: none"> • 요구사항의 타당성
설계	설계 문서	SDD	설계 단계 완료 후	<ul style="list-style-type: none"> • 요구사항의 반영
구현	소스 코드	SRC	구현 단계 완료 후	<ul style="list-style-type: none"> • 설계 결과의 반영

(계속)

개발 단계	품질 관리 대상 항목	식별자	검토 시점	중점 검토 사항
시험	시험 계획서	STP	설계 단계 완료 후	<ul style="list-style-type: none"> • 시험 시나리오 • 시험 대상
	시험 결과서	STR	시험 완료 후	<ul style="list-style-type: none"> • 시험 결과
	사용자 매뉴얼	SUM	시험 완료 후	<ul style="list-style-type: none"> • 설계 결과의 반영

(2) 납품 대상 항목

- 프로젝트 관리 계획서
- 품질 보증 계획서
- 형상 관리 계획서
- 검증 및 확인 계획서
- 요구사항 명세서
- 설계 문서
- 소스 코드
- 시험 계획서 및 시험 결과서
- 사용자 매뉴얼

(3) 품질 관리 활동 및 주요 산출물

품질 관리 활동은 품질 관리 계획 수립, 품질 보증 수행, 품질 통제 수행으로 이루어지며, 세부 활동과 주요 산출물은 다음과 같다.

:: 표 3 | 품질 관리 활동 목록

활동	세부 활동	주요 산출물
품질 관리 계획 수립	품질 관리 계획 수립 및 확정	품질 보증 계획서
품질 보증 수행	품질 보증 활동 수행 및 지원	품질 보증 보고서
품질 통제 수행	품질 관리 결과 보고	<ul style="list-style-type: none"> 품질 보증 평가 계획서 품질 보증 평가 결과 보고서

역할 및 책임

인터넷 쇼핑몰 프로젝트에 참여하는 품질 관리 관련 조직 및 담당자들의 역할과 책임을 정의한다.

::표 4 | 업무 분장

구분	책임
프로젝트 관리자	<ul style="list-style-type: none"> 프로젝트 품질 관리의 총괄적인 책임
품질 보증 팀	<ul style="list-style-type: none"> 프로젝트의 현황 파악 및 모니터링 품질 검토 회의 참석 품질 관리 체크리스트 준비 및 품질 검증 실시 품질 관리 활동 결과 정리 및 보고
개발 팀	<ul style="list-style-type: none"> 프로젝트의 품질 보증을 위한 활동 실시 검토 회의 준비 및 결과 준비/보고

6 산출물

목적

이 장에서는 프로젝트의 진행을 위한 최소한의 산출물 목록을 정의한다. 이 산출물 목록은 품질관리의 대상이 된다.

최소 요구 산출물 목록

- 프로젝트 관리 계획서
- 품질 보증 계획서
- 검증 및 확인 계획서
- 형상 관리 계획서
- 요구사항 명세서
- 설계 문서
- 소스 코드
- 시험 계획서
- 시험 결과서
- 사용자 매뉴얼

7 표준, 실행, 지침, 매트릭스

목적

프로젝트 품질 관리의 성공적인 수행을 위하여 적용되는 표준, 실행, 지침, 매트릭스에 대하여 설명한다.

소프트웨어 품질 관리 방안

소프트웨어 개발에 요구되는 기본적인 검토 작업을 통해 품질 관리를 수행한다.

- 요구사항 검토
- 기본 설계 검토
- 상세 설계 검토
- 검증 및 확인 계획 검토
- 형상 관리 계획 검토

(1) 소프트웨어 표준화 요건

- 국내 및 국제 표준 기술을 수용하도록 한다.
- 업무 분야별 상호 연계성 확보, 유사 분야 간 정보 항목의 중복 개발 및 관리 방지, 정보 항목 간 효율적 연계가 가능하도록 한다.
- 연계할 시스템 간 정보 교환에 관련된 데이터 코드 및 정보 처리 제반 기술을 표준화한다.

(2) 문서 작성 시의 표준화 요건

- 소프트웨어 개발 시 작성해야 할 문서들은 정해진 표준에 따라 작성하도록 한다.
- 작성된 문서의 검토, 승인, 변경, 보관 및 폐기에 관한 절차도 미리 정하도록 한다.

(3) 소프트웨어 개발 시의 표준화 요건

- 개발과정에서의 표준을 소프트웨어 개발을 시작하기 전에 문서화하여 개발자들에게 전달하도록 한다.
- 개발된 프로그램에 대한 검토 또는 검사 시 작성된 표준을 참고하여 진행하도록 한다.
- 만일 개발된 프로그램이 표준과 다를 경우 개발자에게 수정을 요구하여 표준에 맞는 프로그램이 되도록 한다.

8 소프트웨어 품질 보증 활동

소프트웨어 품질 보증 활동은 프로젝트 프로세스 개발, 단계별 산출물 검토, 고객의 합동 기술 검토 참여 및 프로젝트 심사 등의 작업으로 이루어져 있다. 개발사에서는 소프트웨어 품질 보증활동의 독립성을 위해 개발 조직과 독립된 품질 보증 담당자를 선임하여 평가 활동을 진행한다.

(1) 소프트웨어 프로젝트 프로세스 개발

- 개발사의 품질 보증 담당자는 프로젝트의 활동 및 작업 산출물 작성의 기본이 되는 프로젝트 개발 프로세스를 작성한다.
- 개발사의 품질 보증 담당자는 소프트웨어 표준 프로세스의 각 지침에 정의되어 있지 않은 사항에 대해서 프로젝트 팀의 상황에 따라 추가로 정의할 수 있다.

(2) 산출물 검토 및 고객의 기술 검토 참여

- 개발사의 품질 보증 담당자는 필요한 경우, 산출물 검토 또는 고객이 참가하는 기술 검토에 참여하여 아래와 같은 사항을 검토한다.
- 소프트웨어 개발 프로세스 준수 여부
- 작업 산출물에 대한 적합성

(3) 소프트웨어 프로젝트 심사

- 개발사의 품질 보증 조직은 다음과 같은 프로세스 및 작업 산출물의 적합성에 대한 심사를 실시한다.
- 개발 프로세스: 소프트웨어 요구 분석, 소프트웨어 설계, 소프트웨어 구현 및 단위 시험, 소프트웨어 통합 및 인수 시험 등
- 관리/지원 프로세스: 소프트웨어 검증 및 확인 관리, 소프트웨어 형상 관리

소프트웨어 시험**(1) 시험 프로세스 및 산출물**

신규 개발되는 인터넷 쇼핑몰을 대상으로 소프트웨어 품질과 신뢰성 향상 서비스 제공을 위한 시험을 실시한다.

(2) 시험 평가 수행 단계

이 프로젝트에서는 인터넷 쇼핑몰에 대해서 단위 시험과 통합 시험을 실시하며 시스템이 설치될 실제 환경과 유사한 가상 환경에서 사용자 시험을 실시한다.

단계	내용
단위 시험	<ul style="list-style-type: none"> • 모듈의 무결성 입증 • 원시 코드를 대상으로 하나의 모듈이 정상적으로 그 기능을 수행하는지 여부를 시험 • 단위 모듈에 대한 개발 종료 시점에 시험 실시 • 개발자 본인이 단위 모듈에 대하여 시험 실시 • 시험 결과에 대해서 시험 결과 보고서 작성
통합 시험	<ul style="list-style-type: none"> • 단위 모듈에서 시스템으로 단계적으로 통합하며 시험 수행 • 시험 계획에 따라 완성된 시스템을 시험하고, 디버깅하여 시스템이 설계 문서대로 수행되는지 확인 • 시험 결과에 대해서 시험 결과 보고서 작성
시스템 시험	<ul style="list-style-type: none"> • 장애 및 복구 시험 • 동시 사용자를 고려한 과부하 시험(자동화 도구 사용) • 다수의 동시 사용자가 트랜잭션이 발생했을 때 기능이 정상적으로 동작하는지 시험 • 시험 결과에 대해서 시험 결과 보고서 작성
인수 시험	<ul style="list-style-type: none"> • 사용자가 개발 환경에서 수행하는 알파 시험 • 사용자가 실제 운영과 동일한 환경에서 수행하는 베타 시험 • 실제 운영 상황에서 실제 사용자가 수행하는 시험으로 운영상의 문제점을 도출하여 운영에 대비 • 시험 결과에 대해서 시험 결과 보고서 작성

외부 감리

이 프로젝트가 효과적으로 잘 진행되고 있는지 감독하고 관리하도록 외부 감리를 실시한다. 감리의 초점은 개발 프로세스 및 산출물에 대한 품질 보증 활동이 품질 보증 계획대로 수행되었는지 객관적/종합적 점검과 평가를 받고 개선 사항을 도출하여 품질 높은 시스템을 구축한다. 감리는 각 단계별로 수행해야 하나 일정상 기본적으로 총 2회, 중간 감리와 최종 감리로 구분하여 진행하도록 한다.

(1차: 설계 문서 완성, 2차: 수락 시험 완료 후 2주 내)

회차	구분	감리 대상 산출물	공통 감리 대상
1차	중간 감리 (분석 및 설계 감리)	<ul style="list-style-type: none"> • 프로젝트 관리 계획서 • 품질 보증 계획서 • 형상 관리 계획서 • 검증 및 확인 계획서 • 요구사항 명세서 • 설계 문서 	<ul style="list-style-type: none"> • 요구사항 관리 현황 • 형상 관리 현황 • 확인 및 검증 관리 현황 • 품질 보증 현황
2차	최종 감리 (구현 및 시험 감리)	<ul style="list-style-type: none"> • 소스 코드 • 시험 계획서 • 시험 결과서 	

수정 활동

(1) 결함 보고 및 등록

시스템 릴리즈 후 소프트웨어에 대한 결함이 발견되거나 변경 요청이 접수되면 이를 변경 관리 대장에 등록하고 관리한다. 결함 보고 및 등록은 다음과 같은 절차로 진행한다.

절차	내용
결함 발생 보고	<ul style="list-style-type: none"> 시스템 운영 중 결함이 발견되면 결함 발견자는 소프트웨어 변경 관리자에게 결함을 알린다.
결함 접수 등록	<ul style="list-style-type: none"> 변경 관리자는 결함 접수 내용을 소프트웨어 변경 관리 대장에 기록한다.
결함 내용 검토	<ul style="list-style-type: none"> 변경 관리자는 보고된 결함 내용에 대해 검토하고, 접수 및 반송 여부, 형상 항목 변경 필요 여부를 결정한다. 변경 관리자는 결함을 검토한 결과가 반송일 경우, 결함 발견자에게 그 사유를 통보하고 조치를 종료한다. 변경 관리자는 검토 결과 형상 항목 변경이 필요하다고 판단되는 경우 전문가에게 변경 요청서를 작성하여 영향 평가를 의뢰한다.

(2) 결함 조치 절차

접수된 변경 요청서는 다음과 같은 절차에 따라 심사되며, 승인된 변경은 다음과 같은 절차로 진행된다.

절차	내용
변경 요청서 분석 및 변경 영향 평가	<ul style="list-style-type: none"> 변경 관리자는 제출된 형상 변경 요청서의 타당성과 변경을 위한 비용, 시간 등 변경의 영향을 분석 및 검토한다. 분석 및 영향 평가가 완료되면 형상통제위원회에 분석 보고서를 제출한다. 제출된 변경 요청서 분석 및 검토 중 요청자의 잘못된 지식이나 착각에 기인한 변경 요청서가 있을 경우 해당 변경 요청서를 반려하고 그 이유를 제출자에게 설명한다.
변경 심사	<ul style="list-style-type: none"> 형상통제위원회는 제출된 분석 보고서를 바탕으로 해당 변경의 승인 여부를 심사한다. 변경을 승인할 경우, 각각의 형상 항목의 변경을 위한 변경 조치자를 선정하고 그 사실을 변경 조치자에게 공지한다. 변경의 승인 여부를 결정하지 못했을 경우에는 결정을 연기하고, 변경 관리자에게 심사를 위해 필요한 추가적인 분석과 자료를 요청하고, 해당 변경 요청서를 보관한다. 변경이 반려될 경우 반려된 사실과 이유를 변경 요청자에게 전달한다.
형상 항목 변경	<ul style="list-style-type: none"> 형상 관리 담당자는 변경 조치자가 형상 항목을 변경할 수 있도록 형상 항목을 체크아웃하여 변경 조치자에게 전달한다. 형상 항목 변경 조치자는 소프트웨어 형상 변경 조치 계획서를 작성하여 보고한다. 변경 조치자는 전달받은 형상 항목을 변경 조치 계획서에 따라 변경한다. 만일 변경해야 하는 형상 항목이 여러 개고, 각각의 형상 항목이 선행 관계를 갖는 경우, 선행 작업과 후행 작업 사이의 연관성에 모순이 생기지 않도록 확인한다.

(계속)

절차	내용
형상 변경 조치 결과 검토 및 승인	<ul style="list-style-type: none"> • 변경 조치자는 변경 조치가 완료되면 변경 조치 결과서를 작성하여 형상 관리 담당자에게 보고한다. • 변경이 완료된 형상 항목은 다시 정해진 결재과정을 거친다. • 결재가 완료된 형상 항목은 형상 관리 담당자에 의해 형상 관리 서버에 체크인되어 새로운 기준선(baseline) 문서가 되고 리비전(revision)이 증가한다. • 형상 관리 담당자는 새로운 기준선 형상 항목 등록을 전파한다.

공동 기술, 관리 검토

(1) 공동 기술 검토

구분	검토 활동
요구사항 검토 회의	<ul style="list-style-type: none"> • 시스템 전반에 대한 개념 검토 • 요구사항 명세서 검토
기본 설계 검토 회의	<ul style="list-style-type: none"> • 요구사항 명세서에 기초한 기본 설계 문서의 합당성 검토 • 기술적 이슈 사항 검토 및 결정
상세 설계 검토 회의	<ul style="list-style-type: none"> • 기본 설계 문서에 기초한 모듈의 상세 설계 사항 검토 • 기술적 이슈 사항 검토 및 결정
시험 준비 검토 회의	<ul style="list-style-type: none"> • 시험 계획서, 시험 설계서 및 시험 시나리오 검토 • 시험 항목 검토 및 결정
시연회	<ul style="list-style-type: none"> • 인수 시험 종료 후 시연회 실시
수시 검토 회의	<ul style="list-style-type: none"> • 주요 불확실성 및 이슈 사항 관리 및 확정

(2) 공동 관리 검토

구분	검토 활동
사업 착수 회의	<ul style="list-style-type: none"> • 전체 개발 및 관리 계획 검토 • 주요 이슈 사항 검토
요구사항 검토 회의	<ul style="list-style-type: none"> • 진행 사항 및 일정 검토 • 주요 이슈 사항 검토
기본 설계 검토 회의	<ul style="list-style-type: none"> • 진행 사항 및 일정 검토 • 주요 이슈 사항 검토
상세 설계 검토 회의	<ul style="list-style-type: none"> • 진행 사항 및 일정 검토 • 주요 이슈 사항 검토
시험 준비 검토 회의	<ul style="list-style-type: none"> • 진행 사항 및 일정 검토 • 주요 이슈 사항 검토
시험 평가 검토 회의	<ul style="list-style-type: none"> • 진행 사항 및 일정 검토 • 인수 시험 방안 검토 • 주요 이슈 사항 검토
수시 검토 회의	<ul style="list-style-type: none"> • 주요 불확실성 및 이슈 사항 관리 및 확정

9 주요 일정 계획

사업 추진 일정

[illegible]

::그림 2 | 사업 추진 일정표

품질 보증 세부 일정 계획

- 기술 검토회: 각 개발 단계 완료(품질 보증 대상 항목 완성) 후 시행하며, 필요에 따라 프로젝트 관리자에 의하여 추가적으로 시행한다.
- 정기 보고: 매월 프로젝트 품질 보증 현황을 보고한다.
- 시정 조치: 시정 조치가 내려질 경우 시정 조치 계획을 수립한다. 수립된 계획에 따라 프로젝트 시정 조치를 시행한다. 시행 조치가 시행 완료된 후 결과를 보고한다.



Appendix

G

인터넷 쇼핑몰 형상 관리 계획서

1. 목적
2. 범위
3. 핵심 용어 정의 및 약어 용어 | 약어
4. 참고 문헌
5. 형상 관리 조직 및 책임 조직 | 책임 | 적용 가능한 정책, 지시 및 절차
6. 형상 관리 활동 형상 식별 | 형상 통제 | 형상 상태 기록 및 보고 | 형상 감사 및 검토
7. 형상 관리 일정
8. 형상 관리 자원 도구 | 기법 | 요원 | 훈련
9. 형상 관리 계획 유지보수

문서 번호: SEIshop-SCMP

인터넷 쇼핑몰 프로젝트

소프트웨어 형상 관리 계획서(SCMP) (Software Configuration Management Plan)

Rev. 1.0

Approval :

_____	_____/_____/_____
_____	_____/_____/_____
_____	_____/_____/_____

Date

개정 현황					
V1.00	2013. 7. 10	최초 작성	박영수		
개정 번호	일자	변경 사유	작성	검토	승인

1 목적

인터넷 쇼핑몰 시스템 개발 과정 및 운영 유지보수 과정에서 변화하는 소프트웨어의 모습을 가시화하고 그 변경 내용을 체계적이고 일관성 있게 수용하도록 하여 시스템의 품질을 보증한다. 이를 위해 이 계획서는 형상 관리에 참여하는 조직 및 역할을 정의하고 형상 관리 프로세스 및 일정, 자원에 대한 계획을 수립한다.

2 범위

인터넷 쇼핑몰 시스템 형상 관리는 개발 산출물, 관리 산출물, 프로그램 소스, 요구사항에 대하여 적용한다. 형상 관리는 형상 식별, 형상 통제, 형상 보고, 형상 감사 등으로 구성한다.

③ 핵심 용어 정의 및 약어

용어

- 릴리스: 승인된 버전의 형식적인 통지 및 배포(TTA). 시스템 릴리스는 고객에게 배포된 버전으로서 각각의 릴리스는 새로운 기능을 포함하거나 다른 하드웨어 플랫폼을 염두에 둔다.
- 버전: 소프트웨어 형상 항목의 전체 컴파일 또는 재컴파일과 관련된 소프트웨어 형상 항목의 초기 릴리스(IEEE). 시스템의 버전은 다른 인스턴스들과 구별되는 시스템의 인스턴스이다.
- 리비전: 형상 항목의 버전 관리에 사용된다. 이전 리비전도 저장 및 관리되고 사용된다.
- 기준선: (1) 공식적인 검토 및 합의가 끝난 명세(서) 또는 제품을 말하며 개발을 위한 기준이 되고, 공식적인 변경 절차를 통해서만 변경할 수 있다. (2) 형상 항목의 생명주기 동안의 특정한 시점에 미디어에 상관없이 정형적으로 설계되고 확정되어 공식적으로 승인된 구성 항목의 버전이다. 기존의 기준선과 이에 대해 승인된 변화가 추가된 것이 현재의 구성 요소가 된다.
- 형상: (1) 컴퓨터 시스템의 배열 또는 본질, 수 그리고 기능 단위들의 주요한 특성들에 의해서 정의된 네트워크. 더 상세히 말하면 형상에는 하드웨어 형상과 소프트웨어 형상이 있다. (2) 시스템 또는 시스템 성분의 특별한 설명을 정의하는 구현, 설계 그리고 요구사항, (3) 기술적인 문서화 작업에 나타나고 생산에 수행되는 하드웨어와 소프트웨어의 기능적, 물리적 특성(TTA)이다.
- 형상 감사: 요구되는 모든 형상 항목이 만들어졌는지, 현재 형상이 명세화된 요구사항에 일치하는지, 기술적 문서화 작업이 완전하고 정확하게 형상 항목을 서술하는지, 모든 변화가 해결되었는지를 증명하는 과정(TTA)이다.
- 형상 통제: (1) 형상을 공식적으로 설정한 후에 형상 항목의 변화를 수용하는 과정과 평가 승인 또는 비승인의 결정 과정 (2), 체계적 평가, 조정, 승인 또는 불인정, 그리고 공식적인 형상의 설정 후 승인된 형상 항목 변경의 구현(TTA)이다.
- 형상통제위원회(CCB: Configuration Control Board): 제안된 변경 요청 사항을 승인 또는 불인정하고, 승인한 변경의 구현 결과를 확인하고 평가할 수 있는 모임(TTA)이다.
- 형상 확인: (1) 시스템이 형상 항목을 지적하고 형상 항목의 특성을 기록하는 과정, (2) 형상 항목을 정의하는 승인된 문서화 작업, (3) 명세(서), 그림, 연관된 리스트에 나타난 형상 항목에 대해 현재 승인되거나 부분적으로 승인된 문서화 작업과 그것에 참조된 문

서(TTA)이다.

- 형상 항목: (1) 형상 관리를 위한 단위로서 하드웨어 또는 소프트웨어 요소의 집합, (2) 하드웨어나 소프트웨어의 집합 또는 그들의 일부분으로서 최종 사용 함수를 만족하며 형상 관리를 위해 정한다. 항목은 복잡도와 크기에 따라 다양하다. 개발과 초기 생산 단계에서는 형상 항목은 계약 시 직접 언급된 명세(서) 항목이며 운영과 유지보수 단계에서는 수정될 수 있는 항목이다. (3) 최종 사용 기능을 만족하고 주어진 참조 시점에서 유일하게 식별할 수 있는 형상 내의 개체(TTA)이다.

약어

- CCB: Configuration Control Board, 형상통제위원회
- CI: Configuration Item, 형상 항목
- SCM: Software Configuration Management, 소프트웨어 형상 관리

4 참고 문헌

“TTA 소프트웨어 형상 관리 계획 표준”(TTA,IE-828)

“IEEE Standard for Software Configuration management Plans”(IEEE Std 828-1998)

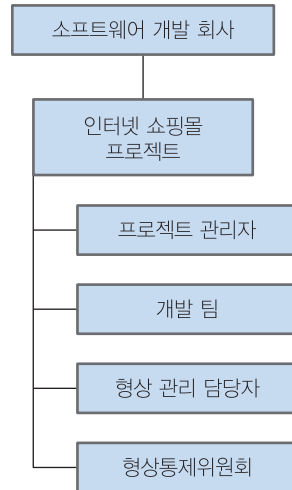
Alexis Leon, *A Guide to Software Configuration Management*, Artec House, 2000.

5 형상 관리 조직 및 책임

다음의 조직에서는 형상 관리에 참여하는 조직을 정의하고, 다음의 책임에서는 각 조직의 책임을 정의한다.

조직

(1) 형상 관리 조직



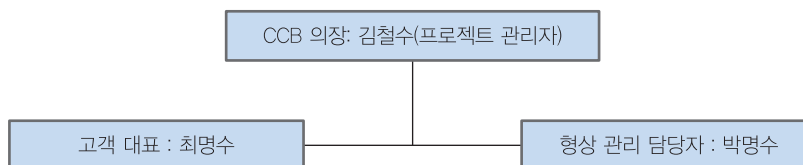
::그림 1 | 형상 관리 조직

형상 관리 조직은 크게 프로젝트 관리자, 형상 관리 담당자, 형상통제위원회, 개발자로 구성한다. 형상 관리 담당자는 프로젝트의 복잡도, 크기 또는 주변 환경에 따라 여러 명이 모인 형상 관리 팀으로 구성될 수도 있고, 형상 관리 담당자 한 명으로 구성될 수도 있다. 인터넷 쇼핑몰 프로젝트의 경우 프로젝트의 규모가 그리 크지 않고 다루어야 할 형상 항목의 수가 많지 않으므로 한 명의 형상 관리 담당자를 선정하여 형상 관리 업무를 수행하도록 한다.

인터넷 쇼핑몰 프로젝트의 형상 관리 담당자는 프로젝트의 구성원으로 프로젝트 내부에서 형상 관리 업무를 수행한다. 형상 관리 담당자는 인터넷 쇼핑몰의 특성에 맞는 형상 관리 활동을 독립적으로 수행할 수 있으나 인터넷 쇼핑몰 프로젝트가 종료된 이후에는 형상 관리 담당자가 없기 때문에 사후 관리에는 어려움이 있을 수 있다.

형상 관리 조직의 각 구성원이 수행하는 역할은 다음의 책임에서 자세하게 기술한다.

(2) 형상통제위원회



::그림 2 | 형상통제위원회

변경 요청이 접수되었을 때 변경 요청을 검토, 승인 또는 거절하기 위해 형상 항목의 변경으로 인해 영향을 받는 사람들로 형상통제위원회(CCB: Configuration Control Board)를 구성한다. 변경이 필요하다고 판단되는 경우 기술 검토를 의뢰받은 분석가는 수정 후 시스템이 어떻게 작동할지, 변경 시 기대 효과 등 분석 결과(변경 영향 분석서)를 형상 관리 담당자에게 보고한다. 분석 결과는 프로젝트 관리자, 고객, 형상 관리 담당자 등으로 구성된 형상통제위원회가 공유할 수 있도록 한다.

형상통제위원회는 프로젝트 관리자, 고객 대표 및 형상 관리 담당자로 이루어져 있는 단일 CCB의 형태를 따른다.

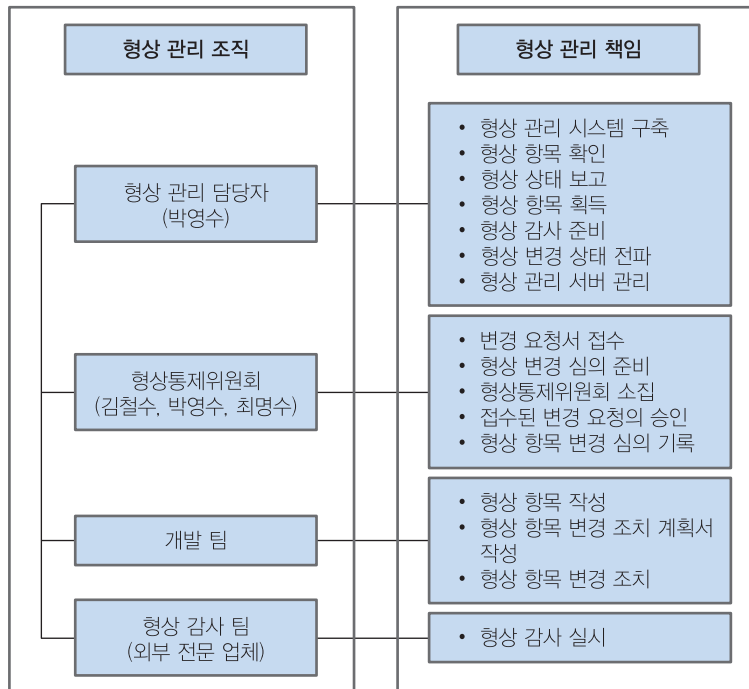
책임

형상 관리 조직의 구성과 함께 각 구성원은 형상 관리 활동에 대한 책임을 부여받는다. 다음 표 1은 형상 관리 조직 구성과 각 구성원의 책임에 대하여 나타낸 표이다.

:: 표 1 | 형상 관리 조직 및 책임

이름	책임
형상 관리 담당자	<ul style="list-style-type: none"> • 형상 관리 시스템 구축 • 형상 항목 확인 • 형상 상태 보고 • 형상 항목 획득 • 형상 감사 준비 • 형상 변경 상태 전파 • 형상 관리 서버 관리
형상통제위원회	<ul style="list-style-type: none"> • 변경 요청서 접수 • 형상 변경 심의 준비(변경 요청서 분석 및 변경 영향 분석) • 형상통제위원회 소집 • 접수된 변경 요청의 승인 • 형상 항목 변경 심의 기록
개발 팀	<ul style="list-style-type: none"> • 형상 항목 작성 • 형상 항목 변경 조치 계획 수립 • 형상 항목 변경 조치
형상 감사 팀	<ul style="list-style-type: none"> • 형상 감사 실시

위 표를 도식화하면 다음과 같은 형상 관리 조직도를 얻을 수 있다.



::그림 3 | 형상 관리 조직도

형상 관리 담당자(박영수)는 형상 관리 업무 전반을 수행하고, 형상 관리 책임자(CMO: Configuration Management Officer)의 역할도 수행한다. 형상통제위원회는 프로젝트 관리자(김철수), 형상 관리 담당자(박영수), 고객 대표(최명수)로 구성되고 형상 통제 위원장은 프로젝트 관리자가 수행하도록 한다.

적용 가능한 정책, 지시 및 절차

인터넷 쇼핑몰의 형상 관리 체계에 대한 외부의 제약 사항(정책, 지시, 절차 등)이 발생할 경우, 프로젝트 관리자 및 형상 관리 담당자는 해당 사항들을 검토 후 이 문서에 기재하고 조직 구성원에게 배포한다. 만일 이러한 제약 사항에 의해 현 형상 관리 체계에 영향 및 효과가 발생하는 경우에는 해당 정보를 이 문서에 기술한다.

6 형상 관리 활동

형상 관리 활동은 형상 식별, 형상 통제, 형상 보고, 형상 감사로 구성되며 각 활동의 세부

활동은 다음과 같다.

형상 식별

소프트웨어 시스템 개발과정에서 산출되는 문서, 코드, 지원 환경의 요소들에 대하여 물리적, 기능적 특성을 식별, 명명한다.

(1) 형상 항목 식별

인터넷 쇼핑몰 프로젝트에서 산출되는 형상 항목들과 각각의 식별자를 다음과 같이 정의한다.

::표 2 | 형상 항목 식별

개발 단계	형상 항목	WBS 산출물	영문 이름	식별자
계획	프로젝트 관리 계획서	프로젝트 관리 계획서	Software Project Management Plan	SPMP
	품질 보증 계획서	품질 보증 계획서	Software Quality Assurance Plan	SQAP
	형상 관리 계획서	형상 관리 계획서	Software Configuration Management Plan	SCMP
	검증 및 확인 계획서	검증 및 확인 계획서	Software Verification and Validation Plan	SWP
분석	요구사항 명세서	유스케이스 명세서 요구사항 명세서	Software Requirements Specification	SRS
설계	설계 문서	화면 설계서 시스템 정의서 테이블 정의서 모듈 명세서	System Design Document	SDD
구현	소스 코드	소스 코드	Source Code	SRC
시험	시험 계획서	시험 계획서	Software Test Plan	STP
	시험 결과서	시험 결과서	Software Test Result	STR
	사용자 매뉴얼	사용자 매뉴얼	Software User Manual	SUM

(2) 형상 항목의 명명

형상 항목의 식별자의 부여 규정은 다음과 같다. 또한 형상 항목의 갱신 과정에서 어떠한 버전 체계를 적용할 것인지 정의한다. 형상 항목의 식별 방법은 명명 규정과 버전 번호 및 문자를 포함할 수 있다. 이 규정은 향후 형상 항목의 저장, 검색, 추천, 재생성 및 분배를 목적으로 한다.

① 문서에 대한 식별 및 버전 부여 규정

- 각 문서의 영문 이름의 두 문자를 추출, 결합하여 명명
- 다른 형상 항목과 중복 시에는 별도의 표기를 추가하여 구별
- 버전은 최초에는 v0.00으로 시작하여 문서의 변경이 일어날 시에 끝에 자리 숫자를 1씩 증가시킨다. (ex. v0.00 → v0.01)
- 문서의 결재(공식화) 때는 Rev를 표시하고 번호를 초기화시킨다. 공식화된 문서가 나중에 변경되었을 경우 끝자리 숫자를 1씩 증가시킨다. (ex. v0.17의 공식화 → Rev0.0, Rev0.0의 변경 → Rev1.0)

② 코드에 대한 식별 및 버전 부여 규정

- 하나의 소스 코드를 형상 항목으로 식별하는 것은 비효율적이기 때문에 소스 코드는 하나의 형상 항목으로 식별하여 관리한다.
- 프로젝트 이름과 소스 코드를 의미하는 약어 SRC를 사용하여 명명한다. (ex. iShop_SRC_v0.00)
- 버전 부여 규정은 문서와 동일하게 적용한다.

(3) 형상 항목의 획득

이 절에서는 형상 항목의 대상이 되는 산출물들의 물리적 저장 및 관리 환경과 문서화 요구사항, 접수 및 검사 요구, 액세스 통제 절차에 대해 서술한다.

① 형상 항목의 물리적 저장 환경

형상 항목으로 식별된 산출물들의 관리를 위해 형상 관리 서버를 구축한다. 형상 관리를 지원하는 소프트웨어를 사용하여 각 형상 항목의 버전 관리를 지원한다. 형상 관리 담당자는 기준선 문서로 공식화될 형상 항목을 정의된 저장소(repository)에 보관한다. 형상 관리 서버에 등록된 형상 항목을 변경하고자 하는 경우 원하는 작업을 수행한 후 변경에 대한 심의를 거쳐 다시 형상 관리 서버에 등록한다. 이러한 과정을 되풀이하면서 각 형상 항목의 리비전

(revision)이 증가한다.

② 문서화 요구사항

형상 항목의 대상이 되는 산출물들은 프로젝트 내에서 정한 규칙에 따라 일관된 양식으로 작성한다. 형상 관리 담당자는 모든 형상 항목에 대해 정해진 양식을 작성하고 조직 구성원들에게 배포한다.

③ 접수 및 검사 요구

형상 항목의 작성(변경)이 완료되면 형상 관리 담당자는 해당 형상 항목이 형상 관리 서버에 등록되어 공식화되기에 적합한지 검사한다. 만일 적합하다면 공식화하고, 부적합하다면 해당 형상 항목이 부적합한 이유와 함께 수정을 요구한다.

④ 액세스 통제 절차

형상 관리 서버에 등록된 형상 항목의 변경을 위해서는 형상 관리 담당자로부터 승인을 얻어야 한다. 형상 관리 담당자는 수정이 필요한 형상 항목을 체크아웃하여 변경 조치자에게 전달한다. 이때 체크아웃된 형상 항목은 지정된 변경 조치자 외에는 수정할 수 없다. 획득한 형상 항목에 대한 작업이 완료되면 해당 형상 항목의 결재 담당자의 검토 및 승인을 거친 뒤에 다시 저장소에 등록한다.

형상 통제

형상 통제 활동은 기준선이 되는 형상 항목들에 대한 변경을 요청, 평가, 승인, 거절하고 실행한다. 변경은 오류 수정과 향상을 포함한다.

(1) 변경 통제

형상 항목의 신규 등록 및 변경은 다음의 절차를 따른다.

① 신규(최초) 형상 항목 등록 절차

- A. 형상 항목의 작성자는 자신이 작성해야 하는 형상 항목을 작성한다.
- B. 작성이 완료된 형상 항목은 최종 승인자의 결재를 받은 후 형상 관리 담당자에게 전달하고, 형상 관리 서버에 새로운 형상 항목으로 등록한다.
- C. 형상 관리 담당자는 신규 형상 항목 등록을 전파한다.

② 형상 변경 요청 및 접수

- A. 소프트웨어 개발 조직 구성원이나 고객은 형상 항목에 대한 변경이 필요하다고 생각될 경우, 소프트웨어 형상 변경 요청서를 작성하여 제출한다.
- B. 변경 관리자(형상 관리 담당자)는 제출된 형상 변경 요청서를 형상 관리 서버에 등록한다.

③ 변경 요청서 분석 및 변경 영향 평가

- A. 변경 관리자는 제출된 형상 변경 요청서의 타당성과 변경을 위한 비용, 시간 등 변경의 영향을 분석 및 검토한다.
- B. 분석 및 영향 평가가 완료되면 형상통제위원회에 분석 보고서를 제출한다.
- C. 제출된 변경 요청서 분석 및 검토 중 요청자의 잘못된 지식이나 착각에 기인한 변경 요청서가 있을 경우 해당 변경 요청서를 반려하고 그 이유를 제출자에게 설명한다.

④ 변경 심사

- A. 형상통제위원회는 제출된 분석 보고서를 바탕으로 해당 변경의 승인 여부를 심사한다.
- B. 변경을 승인할 경우, 각각의 형상 항목의 변경을 위한 변경 조치자를 선정하고 그 사실을 변경 조치자에게 공지한다.
- C. 변경의 승인 여부를 결정하지 못했을 경우에는 결정을 연기하고, 변경 관리자에게 심사를 위해 필요한 추가적인 분석과 자료를 요청하고, 해당 변경 요청서를 보관한다.
- D. 변경이 반려될 경우 반려된 사실과 이유를 변경 요청자에게 전달한다.

⑤ 형상 항목 변경

- A. 형상 관리 담당자는 변경 조치자가 형상 항목을 변경할 수 있도록 형상 항목을 체크아웃하여 변경 조치자에게 전달한다.
- B. 형상 항목 변경 조치자는 소프트웨어 형상 변경 조치 계획서를 작성하여 보고한다.
- C. 변경 조치자는 전달받은 형상 항목을 변경 조치 계획서에 따라 변경한다.
- D. 만일 변경해야 하는 형상 항목이 여러 개고, 각각의 형상 항목이 선행 관계를 갖는 경우, 선행 작업과 후행 작업 사이의 연관성에 모순이 생기지 않도록 확인한다.

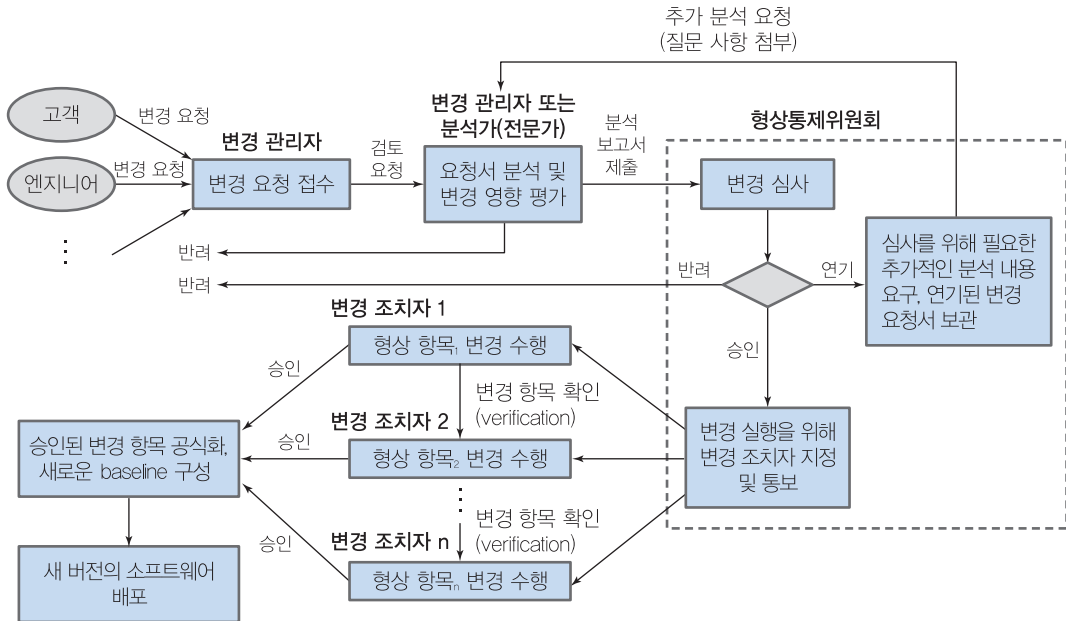
⑥ 형상 변경 조치 결과 검토 및 승인

- A. 변경 조치자는 변경 조치가 완료되면 변경 조치 결과서를 작성하여 형상 관리 담당자에게 보고한다.
- B. 변경이 완료된 형상 항목은 다시 정해진 결재과정을 거친다.

C. 결재가 완료된 형상 항목은 형상 관리 담당자에 의해 형상 관리 서버에 체크인되어 새로운 기준선(Baseline) 문서가 되고 리비전(revision)이 증가한다.

D. 형상 관리 담당자는 새로운 기준선 형상 항목 등록을 전파한다.

아래 그림은 앞에서 설명한 프로세스를 도식화한 것이다. 이를 바탕으로 인터넷 쇼핑몰 프로젝트의 형상 통제 프로세스를 정의한다.



:: 그림 4 | 형상 통제 프로세스

(2) 버전 관리

형상 항목의 버전 관리는 형상 등록 시점을 기준으로 다음과 같이 관리한다.

시점	개정 번호
최초 작성 시	v0,00
개인 변경 시	v0,00에서 뒤의 두 자리 숫자를 1씩 증가(ex. v0,00 → v0,01)
문서의 베이스라인	Rev0,0의 형식으로 초기화. 추후에 변경이 일어날 경우 숫자를 1씩 증가 (ex. v0,17의 공식화 → Rev0,0, Rev0,0의 변경 → Rev1,0)

(3) 사용자 접근 관리

소프트웨어 형상 항목에 대한 접근 관리는 다음과 같다.

사용자	권한
형상 관리 담당자	<ul style="list-style-type: none"> • 형상 관리 서버의 전반적인 사항을 관리한다. • 형상 항목의 체크인, 체크아웃 권한을 갖는다. • 관련자에게 형상 항목의 읽기 및 쓰기 권한을 줄 수 있다.
프로젝트 관리자 및 개발자	<ul style="list-style-type: none"> • 형상 항목의 읽기 권한을 갖는다. • 필요 시에 담당자의 허가를 얻어 쓰기 권한을 일시적으로 갖는다.
기타 프로젝트 관련자	<ul style="list-style-type: none"> • 필요 시에 담당자의 허가를 얻어 형상 항목의 읽기 권한을 갖는다.

형상 상태 기록 및 보고

형상 상태 기록 항목

- 형상 항목 관리 대상: 신규로 식별되거나 형상 통제 활동을 통해 수정된 형상 항목에 대한 식별 번호, 이름, 버전, 기준 일자 등 현재 형상 상태의 기록이다. 형상 관리 담당자가 작성한다.
- 변경 요청서, CCB 회의록: 형상 통제 활동의 결과물로, 형상 항목 변경 시 해당 형상 항목의 이전 상태를 추적할 수 있도록 해당 형상 항목에 대한 변경 이력 및 내역을 기록한다. 변경 요청서는 고객 또는 개발자가 작성하고, CCB 회의록은 CCB 일원이 작성한다.
- 형상 감사 보고서: 형상 감사 결과를 기록한다. 형상 감사 팀에서 작성한다.
- 형상 상태 보고서: 형상 항목의 상태를 기록하는 것으로 형상 관리 담당자가 작성한다.

형상 관리 담당자는 지정된 주기에 따라 기록된 형상 상태를 바탕으로 형상 상태 보고서를 작성한다. 작성된 형상 상태 보고서는 프로젝트 관리자에게 제출한다. 형상 관리 담당자는 보고된 형상 상태 보고서를 프로젝트 참여 인원에게 전파 또는 형상 관리 서버에 저장하여 현재 시점의 최종 형상 상태를 공유한다.

형상 상태 보고서에는 형상 항목에 대해서 무슨 일이 있었는지, 누가 수행했는지, 언제 일어났는지, 다른 어떤 것이 영향을 받았는지에 대한 내용을 기록한다. 형상 항목이 새롭게 작성되거나 수정되면 형상 상태 보고서를 작성한다. 또한 형상 감사가 시행될 때마다 그 결과를 형상 감사 보고서로 기록하고, 그 결과물은 형상 관리 서버에 기록하여 개발자나 지원 팀에서 확인할 수 있도록 한다.

형상 감사 및 검토

(1) 형상 감사 준비

- 형상 관리 담당자는 프로젝트 관리자와 함께 형상 감사자를 선정한다.
- 형상 감사자는 소프트웨어 형상 기준선 체크리스트를 작성하고, 해당 기준선의 형상 항목을 수집한다.
- 형상 관리 담당자는 형상 감사자를 대상으로 형상 기준선 설정 내역을 설명하고, 형상 상태 보고서를 비롯한 형상 기준선 감사에 필요한 제반 정보 및 자료를 제공한다.

(2) 형상 감사 실시

- 소프트웨어 형상 기준선 체크리스트에 따라 형상 감사자는 형상 감사를 실시한다.
- 감사 수행 중 결함을 발견한 경우에 적절한 조치를 취한다.

(3) 형상 감사 기준선 결과 보고 및 사후 관리

- 형상 감사 수행 후 형상 감사 보고서를 작성하여 프로젝트 관리자에게 보고한다.
- 프로젝트 관리자는 감사로 발견된 결함에 대한 조치를 취하고 관리한다.

7 형상 관리 일정

주	개발 단계				완료 단계			
1	계획	발급						발급 계획서
2								비밀한위 발급 계획서
3								공조 발급 계획서, 진검 및 확인 계획서
4								여다사공 공제서
5								
6								
7								
8								
9								
10								결계 공제서
11								
12								
13								
14								
15								
16								
17								
18								선시 면비
19								시공 계획서
20								시공 계획서
21								
22								사여다공 공제서

::그림 5 | 형상 관리 일정표

각각의 개발 단계가 끝나면 형상 감사를 실시한다. 각각의 형상 감사는 해당 단계에서 작성되기로 한 산출물 및 형상 항목이 정확히 작성되었는지 확인하고, 각각의 산출물 및 형상 항목이 논리적인 일관성 및 사용자의 요구사항을 정확히 반영하였는지 확인할 수 있도록 한다. 또한 형상 관리 활동을 수행함에 있어서 형상 관리 계획서에 명시되어 있는 대로 수행되고 있는지, 정확한 보고 체계를 준수하고 있는지, CCB가 제 역할을 수행하는지를 확인하도록 한다. 만약 형상 감사 중에 부적절한 부분이 발견되었다면 형상 감사 보고서에 기록하고 보고하여 프로젝트 참여 인원들이 확인하고 수정할 수 있도록 한다.

개발 단계 수행 중에 식별된 형상 항목에 대한 모든 내용(신규 등록, 변경 요청, 변경 심사, 변경 수행 내역, 현재 상태 등)을 기록하여 모든 프로젝트 참가 인원이 확인할 수 있도록 한다.

8 형상 관리 자원

인터넷 쇼핑몰 프로젝트의 형상 관리 활동을 위해 필요한 소프트웨어 도구, 기법, 자원, 훈련 정보를 아래와 같이 정리한다.

도구(소프트웨어): Beeskit Integrated Edition, Version 1.0.0

- 형상 항목 저장 및 버전 관리 지원
- 문서, 소스 코드 관리
- 각 개발자의 개발 공간 및 환경을 지원하고 타 개발자와 작업 환경을 공유할 수 있도록 지원

기법

개발과정에서 산출된 문서 또는 소스 코드 등에 대한 버전 관리는 Beeskit Integrated Edition을 활용하고, 변경 통제와 관련된 활동(변경 요청, 변경 심의, 변경 검토 및 승인 등)은 조직 내 책임 규정에 의거하여 관리한다. 형상 항목에 대한 체크아웃이 발생하여 변경이 진행 중인 경우, 변경 조치자를 제외한 나머지 사용자들은 해당 형상 항목을 변경할 수 없다. 이는 동시 작업으로 인한 업무의 중복, 충돌 문제들을 방지하기 위함으로, 이를 적용함으로써 형상 항목의 변경에 대한 일관성 및 업무의 효율성을 유지할 수 있다.

요원

형상 관리 담당자가 전반적인 형상 관리 활동을 지휘한다. 프로젝트 관리자와 고객 대표는 형상 항목의 변경 요청의 심의 등의 업무를 지원한다. 일반적인 프로젝트 구성원들은 조직 내 책임 규정에 의거하여 활동하며, 형상 항목의 등록이나 변경을 위해서는 형상 관리 담당자의 승인을 얻는 등 정해진 절차를 거쳐야만 한다.

훈련

형상 항목의 식별, 획득 및 변경에 대한 전반적인 내용을 프로젝트 구성원에게 훈련시킨다. 특히, 변경 통제 과정을 숙지시켜 동시 작업 및 그 외의 문제 발생을 예방하고 형상 관리 정보의 공유가 체계적으로 이루어질 수 있도록 체계를 확립하며 내용을 숙지시킨다.

9 형상 관리 계획 유지보수

형상 관리 담당자는 프로젝트 생명주기 동안 지속적으로 수행되고 보증되어야 하는 형상 관리 계획에 대해 책임을 갖는다. 형상 관리 계획은 프로젝트 수행 초기인 계획 단계에서 수립되며, 확정된 형상 관리 계획서는 프로젝트에 참여한 모든 조직 구성원이 숙지하고 이행할 수 있도록 한다. 형상 관리 계획의 변경이 필요한 경우 프로젝트 관리자와 형상 관리 담당자는 그에 따른 영향 정도를 검토하여 변경 여부를 결정한다. 형상 관리 계획이 변경되면 변경된 형상 관리 계획서 내용 등 변경 관련 정보를 프로젝트에 참여한 모든 조직 구성원에게 즉시 배포하여 확인 및 숙지할 수 있도록 한다.



Appendix



인터넷 쇼핑몰 검증 및 확인 계획서

1. 목적
2. 참고 문서
3. 정의
4. V&V 개요 조직 | 기본 종합 일정 | 자원 요약 | 책임 | 도구, 기법, 방법론
5. 생명주기 V&V V&V 관리 | 계획 단계 V&V | 분석 단계 V&V | 설계 단계 V&V | 구현 단계 V&V | 시험 단계 V&V | 설치 및 점검 단계 V&V | 운영 및 유지보수 V&V
6. 보고 요구되는 보고서
7. V&V 감독 절차 예외 사항 보고 및 해결

문서 번호: SEIshop-SVVP

인터넷 쇼핑몰 프로젝트

소프트웨어 검증 및 확인 계획서(SVVP) Software Verification and Validation Plan

Rev. 1.0

Approval :

_____	_____/_____/_____
_____	_____/_____/_____
_____	_____/_____/_____

Date

개정 현황					
0.8	2013-07-23	시험 단계 추적 가능성 분석 추가	박영수		
0.7	2013-07-20	단계별 V&V 작업 명세 방법 변경	박영수		
0.6	2013-07-18	설계 V&V 통합	박영수		
0.5	2013-07-17	V&V 계획서 초안 작성 완료	박영수		
0.4	2013-07-15	상세 설계 및 구현 단계 V&V 활동 작성	박영수		
0.3	2013-07-12	추적 가능성 매트릭스 추가	박영수		
0.2	2013-07-10	개발 조직 및 역할 수정	박영수		
0.1	2013-07-05	최초 작성	박영수		
개정 번호	일자	변경 사유	작성	검토	승인

1 목적

이 문서는 SE 인터넷 쇼핑물 프로젝트의 소프트웨어 검증 및 확인에 대한 계획을 나타내기 위해 작성한다. 이 계획서는 소프트웨어에 대한 사용자 요구사항이 제대로 만족될 수 있도록 보장하기 위한 각종 절차와 활동들을 기술한다. 계획 단계에서부터 요구 분석 단계, 설계 단계, 구현 단계, 시험 단계, 유지보수 단계에 이르는 각 생명주기에서 이루어지는 검증 및 확인 활동들을 기술하며, 각 활동에 대한 보고서 작성 방안을 계획한다. 소프트웨어의 검증 및 확인 활동을 통해 인터넷 쇼핑물 시스템에 대한 사용자 요구사항들이 제대로 만족되었는지 여부를 판단할 수 있도록 한다.

이 계획서는 SE 인터넷 쇼핑몰 개발 프로젝트의 품질 관리 팀에 의해서 작성하며, IEEE의 소프트웨어 검증 및 확인 계획서 지침(IEEE Std 1012-1986)과 한국정보통신기술협회(TTA) 소프트웨어 검증 및 확인 계획서 지침(TTAS,IE-1059)을 준수한다. 소프트웨어 검증 및 확인과 관련된 조직, 일정, 기법 등과 함께 프로젝트 생명주기별 검증 및 확인을 위한 활동들을 기술한다.

2 참고 문서

이 계획서는 아래와 같은 문서들을 토대로 작성한다.

“IEEE Standard for Software Verification and Validation Plans”(IEEE Std 1012-1986)

“소프트웨어 검증 및 확인 계획서 지침”(TTAS,IE 1059)

“프로젝트 관리 계획서”(SEIshop-SPMP)

“소프트웨어 품질 보증 계획서”(SEIshop-SQAP)

“소프트웨어 형상 관리 계획서”(SEIshop-SCMP)

“IEEE Standard for Software and System Test Documentation”(IEEE Std 829-2008)

“소프트웨어 시험 문서화 표준”(TTAS,IE-829)

“소프트웨어 시험 계획서”(SEIshop-STP)

3 정의

이 계획서에서 사용된 용어, 두문자어, 약어에 대한 정의를 다음과 같이 기술한다.

::표 1 | Acronyms

용어 · 두문자어 · 약어	정의
V&V	Verification and Validation
SWVP	Software Verification and Validation Plan
SVVR	Software Verification and Validation Report
SDP	Software Development Plan
SRS	Software Requirement Specification
SDD	Software Design Document
IDD	Interface Design Document

(계속)

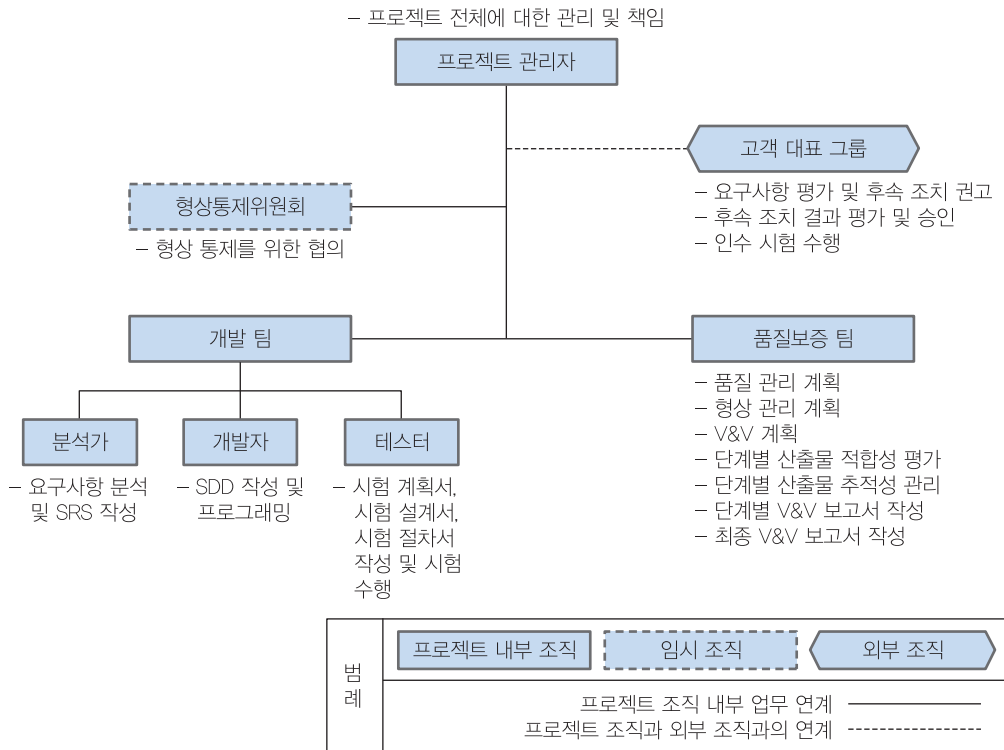
용어 · 두문자어 · 약어	정의
SRC	Source Code
UTP	Unit Test Plan
ITP	Integration Test Plan
STP	System Test Plan
ATP	Acceptance Test Plan
SPMP	Software Project Management Plan
SCMP	Software Configuration Management Plan
PM	Project Manager
QAM	Quality Assurance Manager
TM	Test Manager
CMO	Configuration Management Officer
RMP	Risk Management Plan
PDR	Preliminary Design Review
CDR	Critical Design Review

4 V&V 개요

이 장은 SE 인터넷 쇼핑몰 시스템의 V&V 활동을 위한 조직 구성, 일정, 자원, 조직 구성원의 책임, 관련 도구, 기법, 방법론에 대한 내용을 기술한다.

조직

SE 인터넷 쇼핑몰 프로젝트와 관련된 조직 구성도는 그림 1과 같다. 각 팀 또는 담당자들에 대한 책임은 다음에서 보다 상세히 설명하고 있다.



::그림 1 | 프로젝트 조직 구성도

기본 종합 일정

각 생명주기 단계별 V&V 활동에 대한 일정은 다음과 같다. 각 생명주기 단계마다 품질 활동에 대한 일정이 포함되어 있다. 계획 단계에서는 품질 활동 계획 기간에 V&V 계획서 작성 이 이루어진다. 분석, 설계, 구현, 시험, 설치 단계에서는 모든 작업이 완료된 후 품질 보증 활동에 대한 보고가 이루어지며, 이 기간 동안 V&V 보고서가 작성된다. 일정표에서 검은색 막대로 표시된 기간에 V&V 계획서 및 보고서가 작성된다. 표 상단의 숫자 일정 단위는 일주일이다.

단 계	작업 패키지	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22
계 획	프로젝트 관리 지원 계획																						
	품질 활동 계획																						
	개발 계획서 작성																						
분 석	유스케이스 모델링																						
	요구사항 분석																						
	프로토타입 구축 및 검증																						
	분석 단계 품질 보증 활동 보고																						
설 계	사용자 인터페 이스 설계																						
	시스템 구조 설계																						
	데이터베이스 설계																						
	프로시저 설계																						
	설계 명세서 작성																						
	설계 단계 품질 보증 활동 보고																						
	프로그램 구현																						
구 현	구현 단계 품질 보증 활동 보고																						
	시험 계획서 작성																						
	시험 수행 및 결과서 작성																						
시 험	시험 단계 품질 보증 활동 보고																						
	DB 구축 및 소 프트웨어 설치																						
	설치 확인 및 형상 감사																						

::그림 2 | 소프트웨어 개발 일정표 및 V&V 작업 일정

자원 요약

소프트웨어 V&V를 위한 인원, 설비, 도구, 재정, 보안, 접근 권한, 문서 통제에 필요한 자원 계획이 마련되어야 한다.

V&V 활동을 수행하는 인원은 품질 관리 팀의 일부로 구성하도록 한다. 품질 관리 팀 산하 V&V 활동을 총괄 운영하는 V&V 담당자를 둔다. V&V 담당자는 단계별 V&V 활동(단계별 산출물 평가, 추적 가능성 분석, 검토, 시험 등)에 대한 보고서를 작성한다. V&V 담당자들이 V&V 보고서에 기록된 문제를 개발 팀이나 다른 구성원들에게 통보할 수 있도록 의사 전달 체계(사내 메신저 프로그램 등)가 별도로 구축되어 있어야 한다.

V&V 활동의 수행과정에서 소요되는 예산(예를 들어, 검토회 개최 시 필요한 비품, 전문가 초청비용 등)은 품질 보증 활동에 책정된 예산의 일부로 한다. V&V의 대상이 되는 산출물들에 대한 보안 및 접근 권한, 문서 통제와 관련된 이슈는 형상 관리 계획서상에 기록된 내용을 따르도록 한다.

책임

(1) 프로젝트 관리자

- SE 인터넷 쇼핑몰의 프로젝트 관리 계획상에 기술된 내용의 진척 상황을 모니터링하며, 프로젝트 진행 과정에서 발생하는 모든 문제들을 통제한다.
- 검토회 등과 같은 검증 및 확인 활동의 제반 사항을 총괄 지휘한다.

(2) 품질 보증 팀(V&V 담당자 포함)

- V&V 계획서를 작성하고 업데이트한다.
- V&V 팀의 활동과 V&V 작업의 일정을 관리한다.
- 요구된 V&V 활동을 수행한다.
- 소프트웨어 V&V 계획서(SVVP)에 명시된 프로세스에 따라 검증 및 확인 활동이 적절히 수행되었는지를 확인한다.
- V&V 활동에 의해 발견된 문제에 대해서 적절한 후속 조치가 이루어질 수 있도록 통보하고, 처리된 후속 조치 결과를 확인한다.
- 각 생명주기 단계별 V&V 보고서를 작성한다.
- 최종 V&V 보고서를 작성한다.

(3) 개발 팀

- 개발 팀의 분석가는 제안 요청서 및 제안서를 기반으로 SRS를 작성한다.
- 개발 팀의 개발자는 SRS를 기반으로 설계 및 구현 작업을 수행한다.
- 개발 팀의 테스터는 시험 계획서, 시험 설계서, 시험 절차를 작성하고, 구성 요소 시험, 통합 시험, 시스템 시험을 수행한다. 시험의 결과는 보고서로 작성한다.
- 개발 팀 전원은 형상 관리 활동에 참여하고, 형상 관리 담당자로 지정된 일부 인원은 형상 관리 활동이 적절히 이루어질 수 있도록 지원한다.
- 소프트웨어 산출물에 대한 평가가 이루어질 수 있도록 평가 대상 산출물들을 제공한다.
- V&V 계획서에 근거하여 실질적인 V&V 임무를 수행한다.

(4) 형상통제위원회

- 형상 항목의 변경과 관련된 통제의 역할을 수행한다.
- 요청된 변경의 영향에 대한 분석 결과를 평가하고, 변경 수행 여부를 결정한다.

(5) 고객 대표 그룹

- SE 인터넷 쇼핑몰 관리자로 구성된 고객 대표 그룹(customer representative group)은 검토회와 같은 V&V 활동에 참석하여 고객의 요구사항을 명확하게 전달할 수 있도록 하며, 검토회에서 발견된 문제에 대한 후속 조치를 권고할 수 있다.
- V&V 담당자와 함께 후속 조치의 결과를 검토한다.
- 사용자 인수 시험을 수행한다.

도구, 기법, 방법론

V&V 활동들의 목적을 달성하기 위해서 선택된 다양한 도구들, 기법들, 방법론들을 기술한다.

(1) 도구

- 문서 작성 도구: MS Word, HWP
- 형상 관리 도구: Beeskit Integrated Edition
- 시험 도구: JUnit
- 프로젝트 관리 도구: Microsoft Project
- 요구사항 관리 및 추적 도구: IBM Rational Doors(사정에 따라 스프레드시트를 이용한 수동 관리로 대체될 수 있음)

(2) 기법 및 방법론

V&V 활동은 다음과 같은 기법에 의해 진행된다.

- 추적 가능성 분석(analysis of traceability): 최종 개발 제품이 적정한 품질 수준을 유지하고 고객의 요구를 만족하는지 확인하기 위해 검증 및 확인 팀은 모든 요구사항이 설계, 구현, 시험사례에 포함되었는지를 확인해야 한다. 추적 가능성 분석은 요구사항, 설계, 코드, 시험 사례들에 대하여 추적하는 것을 의미한다. 분석 단계에서는 식별된 요구사항들이 계획 단계의 제안 요청서 및 제안서로부터 정확하게 유도되었는지를 확인한다. 설계 단계에서는 분석 단계에서 작성된 SRS를 따라 SDD가 정확하게 작성되었는지를 확인한다. 분석 단계에서 작성된 시나리오와 클래스 다이어그램으로부터 설계 단계의 요소들(화면, 상세 클래스, 데이터베이스 테이블 등)이 정확하게 추적되는지 확인해야 한다. 구현 단계에서는 설계 단계에서 작성된 클래스 다이어그램과 시퀀스 다이어그램에 따라 클래스들을 구현하였는지, 클래스 내부의 알고리즘이 설계 단계의 프로시저 설계 내용과 일치하는지 확인한다. 각 요소들에 대한 추적 가능성 여부를 나타내기 위해 추적 가능성 매트릭스를 사용한다.
- 검토(reviews): 모든 납품 산출물들은 작성과 함께 검토의 대상이 된다. 검토는 각 산출물들이 프로젝트의 요구사항을 만족하는지 확인하는 것이다. 검토 프로세스를 통해 승인된 산출물은 형상 통제하에 놓이게 된다. 만약 검토 중에 불일치 사항이 발견되면 이상 보고서를 작성하여 제출한다. 이 보고서에 따라 검토 대상 산출물과 그로 인해 영향을 받는 다른 산출물들도 변경될 수 있다.
- 코드 검사(code inspection): 구현 단계에서 개발된 모듈의 소스 코드가 설계 명세서에 의해 식별된 요구사항을 만족하는지 확인한다. 이러한 분석에서는 두 가지 관점이 포함된다. 첫 번째는 모든 요구된 기능이 포함되었는지, 코드가 표준에 부합하는지 확인하는 것이다. 두 번째는 코드가 하드웨어 및 소프트웨어의 인터페이스와 잘 호환되는지 여부를 확인하는 것이다.
- 시험(testing): 시험은 구성 요소(단위) 시험, 통합 시험, 시스템 시험, 인수 시험으로 구성된다. 프로젝트 계획 단계에서 시험의 전반적인 사항을 계획하고, 분석, 설계, 구현 단계가 완료되는 시점에 각 시험에 대한 설계가 이루어진다. 시험에 대한 절차는 구현 단계에서 작성된다. 시스템 시험, 통합 시험, 구성 요소 시험은 개발 팀 내부 구성원들에 의해 이루어지며, 인수 시험은 고객의 참여를 통해 진행된다.

5 생명주기 V&V

이 장에서는 SE 인터넷 쇼핑물 프로젝트 생명주기 전반에 걸쳐 수행되는 V&V 활동들의 계획에 대해 요약한다. 각 V&V 활동에 대한 입력물과 출력물이 명시되며, 각 활동의 수행방법 및 기준에 대해서 소개한다.

V&V 관리

쇼핑물 프로젝트의 개발 프로세스 전반에 걸쳐 수행되어야 하는 관리 활동들을 계획한다. 관리 V&V 활동에서 필요한 작업과 입/출력 산출물들은 다음과 같다.

(1) V&V 작업

- SVVP 생성
- 기준선 변경 심사
- V&V에 대한 관리 검토
- 관리적 및 기술적 검토 지원

(2) 입력물

모든 생명주기 단계의 V&V 활동에 대한 입력물 및 출력물

(3) 출력물

- SVVP 및 갱신 사항
- 작업 보고서
- 예외 사항 보고서
- V&V 활동 요약 보고서
- V&V 최종 보고서에 대한 권고 사항

계획 단계 V&V

계획 단계에서는 시스템의 개발 동기를 수립한다. 개발될 시스템의 본질을 정의하고 기술적이고 사업적인 제약 조건 내에서 그것의 목적과 위험 요소들을 열거한다. 계획 단계의 V&V 활동에서 필요한 작업과 입/출력 산출물은 다음과 같다.

(1) V&V 작업

- 개념 문서 평가
- 추적 가능성 분석

(2) 입력물

개념 문서(제안 요청서, 제안서), 프로젝트 관리 계획서, 품질 관리 계획서, 형상 관리 계획서, 시험 계획서

(3) 출력물

계획 단계 V&V 작업 보고서, SVVP

분석 단계 V&V

분석 단계에서는 요구사항이 도출, 평가, 분석되며, 이러한 과정에서 고객들은 도출된 요구사항들이 자신들이 제시한 필요와 적절하게 대응되는지 여부를 판단하기 위해 개발 팀과 함께 해야 한다. 분석 단계에서의 V&V의 목적과 관련 활동 정보들을 기술하면 다음과 같다.

(1) V&V 작업

- 소프트웨어 요구사항 추적 가능성 분석
- 소프트웨어 요구사항 평가
- 소프트웨어 요구사항 인터페이스 분석
- 시험 계획 작성 및 검증(시스템 시험, 수락 시험)

(2) 입력물

- 개념 문서
- SRS

(3) 출력물

- 분석 단계 V&V 작업 보고서
- 시스템 시험 계획
- 수락 시험 계획

설계 단계 V&V

설계 단계에서는 소프트웨어 아키텍처, 소프트웨어 구성 요소, 인터페이스, 데이터에 대한 설계가 새롭게 작성된다. 설계의 내용이 요구사항에 맞게 작성되었는지를 검증하는 데 초점을 맞춘다. 설계 단계 V&V 활동을 통해서 설계의 오류를 제거하고 이를 통해 이후의 코드에서 결함이 최소화될 수 있도록 한다. 분석 단계에서 정의한 논리 모델이 설계 모델로 발전되는 과정에서 누락된 요소가 없는지 확인한다.

(1) V&V 작업

- 소프트웨어 설계 추적 가능성 분석
- 소프트웨어 설계 평가
- 시험 계획 작성(구성 요소 시험, 통합 시험)
- 시험 설계 작성(구성 요소 시험, 통합 시험, 시스템 시험, 수락 시험)

(2) 입력물

- SRS
- SDD
- 설계 표준
- 개념 문서

(3) 출력물

- 설계 단계 V&V 작업 보고서
- 시험 계획(구성 요소, 통합)

(4) 시험 설계

구성 요소, 통합, 시스템, 수락

구현 단계 V&V

구현 단계는 소프트웨어 산출물이 설계 문서로부터 생성되고 디버깅되는 생명주기 기간이다. 구현 단계에서 V&V 작업들은 SDD와 코딩 표준을 얼마나 준수하는가에 대해 확인한다. 구현 단계에서의 V&V의 목적은 소스 코드의 품질을 결정하는 것이다.

(1) V&V 작업

- 소스 코드 추적 가능성 분석
- 소스 코드 평가
- 시험 사례 작성(구성 요소 시험, 통합 시험, 시스템 시험, 수락 시험)
- 시험 절차 작성(구성 요소 시험, 통합 시험, 시스템 시험)
- 구성 요소 시험 실행

(2) 입력물

- SRS
- SDD
- 소스 코드
- 코딩 표준
- 개념 문서

(3) 출력물

- 구현 단계 V&V 작업 보고서
- 시험 사례(구성 요소, 통합, 시스템, 수락)
- 시험 절차(구성 요소, 통합, 시스템)

시험 단계 V&V

구현된 코드의 품질을 확인하고, 코드에서 발견될 수 있는 결함을 최소화하기 위해 V&V 작업을 수행한다. 주로 사용자의 기능적 요구사항과 비기능적 요구사항에 대해서 시스템이 만족하고 있는지 여부를 평가한다.

(1) V&V 작업

- 시험 단계 추적 가능성 분석
- 수락 시험 절차 작성
- 시험 실행(통합 시험, 시스템 시험, 수락 시험)

(2) 입력물

- 시험 사례 및 절차

- SRS
- SDD
- 소스 코드 및 실행 가능한 코드

(3) 출력물

- 시험 단계 V&V 작업 보고서
- 수락 시험 절차

설치 및 점검 단계 V&V

소프트웨어 산출물이 운영 환경으로 통합되고, 요구된 대로 제대로 수행되는가를 보장하기 위해 시험한다.

(1) V&V 작업

- 설치 형상 감사
- V&V 최종 보고서 작성

(2) 입력물

- 설치 패키지
- 각 생명주기 단계별 V&V 작업 보고서

(3) 출력물

- 설치 및 점검 단계 V&V 작업 보고서
- V&V 최종 보고서

운영 및 유지보수 V&V

필요에 따라 소프트웨어 산출물을 정정하거나, 변화하는 요구사항에 대해 반응하는 각종 활동들에 대해서 검증하고 확인한다.

(1) V&V 작업

- SVVP 개정

- 예외 사항 평가
- 제안된 변경 평가
- 단계 작업 반복

(2) 입력물

- SVVP, 새로운 제약 사항
- 제안된 변경 사항
- 설치 패키지
- 운영 절차
- 사용자 문서
- 개념 문서

(3) 출력물

운영 및 유지보수 단계 V&V 작업 보고서

6 보고

생명주기 전반에 걸쳐 V&V 보고서가 작성되어야 하며, 각 보고서에는 V&V 활동의 상태와 발견된 모든 사항을 기술한다.

요구되는 보고서

각 생명주기 단계별 수행되는 V&V 활동에 대해서 다음과 같은 보고서들이 작성되어야 한다.

- 예외 사항 보고서: 식별된 예외 사항을 기록하고, 빠른 조치를 위해 개발자들에게 전달해야 하는 보고서이다. V&V 담당자는 발견된 예외 사항에 대해서 품질 보증 담당자 및 프로젝트 관리자와 협의하여 후속 조치의 일정, 비용을 예측하고, 개발자들에게 관련 내용을 통보하여 후속 조치가 적절히 이루어질 수 있도록 관리한다.
- V&V 작업 보고서: 각 생명주기 단계에서 수행된 V&V 활동을 증명하기 위해 작성한다. V&V 활동들에 대한 결과를 상세히 기술한다.
- V&V 단계 요약 보고서: 각 생명주기 단계별로 각 조직에 의해 수행된 V&V 활동들의

결과물들을 요약하고 통합한다.

- V&V 최종 보고서: 예외 사항, V&V 작업, 모든 V&V 단계 보고서의 결과물을 설치 및 점검 단계에서 요약하고 통합한다.

7 V&V 감독 절차

예외 사항 보고 및 해결

각 V&V 작업의 책임자들은 발견된 예외 사항을 반드시 기록하여 V&V 담당자에게 전달해야 한다. V&V 담당자는 각 생명주기 단계별로 수행되는 V&V 작업에 의해서 발견된 예외 사항을 검토하고, 예외 사항을 처리하기 위해 각 단계별 산출물 작성자(개발 팀 구성원)들에게 반드시 알려야 한다. 예외 사항에 대해서 후속 조치가 이루어지면 개발 팀 구성원들은 개선의 결과를 다시 V&V 담당자에게 전달한다. V&V 담당자는 후속 조치 결과를 분석하여 다음 단계로의 진입 여부를 결정한다.



Appendix

인터넷 쇼핑몰 원가 관리(기능 점수 적용 사례)

1. COCOMO 모델
2. 기능 점수 기반 비용 측정 기능 점수 항목 | 기능 점수 산정 절차 | 데이터 기능 점수 측정
트랜잭션 기능 점수 측정
3. 프로세스 기반 공수 추정 모델
4. 기능 점수 산정 방식 간이법 적용 | 정규법 적용

인터넷 쇼핑몰 원가 관리(기능 점수 적용 사례)

1 COCOMO 모델

소프트웨어의 경우 프로젝트의 규모를 측정하기 위해 전통적으로 널리 사용되는 단위는 프로그램이 몇 라인(줄)인가를 나타내는 LOC(Lines Of Code)이다. 이 방법은 1981년 보엠(B. Boehm)에 의해 소개된 COCOMO(Constructive COst MOdel)로 대표되며, 프로그램의 규모인 LOC를 추정하고 이를 준비된 식에 대입하여 소요되는 기간과 인원을 구하여 개발 비용을 예측하는 대표적인 하향식 모델이다. COCOMO 모델은 비교적 작은 규모의 프로젝트 기록을 통계 분석하여 얻은 결과를 반영한 모델이며 중소 규모 소프트웨어 프로젝트 비용 추정에 적합하다.

COCOMO 모델은 첫 번째 기본 단계에서 개발 유형(프로젝트 복잡도)을 바탕으로 기본 공식에 적용하여 대략적인 노력 추정을 계산하고, 두 번째 단계에서는 이 추정치에 원가 유발 요인인 다수의 비용 인자(cost driver)를 적용하여 예상 노력을 보정한다.

COCOMO 모델은 기본 단계에서 다음과 같이 표현된다.

$$\text{예상되는 노력: } E = c \times (KDSI)^k$$

여기서 노력은 한 사람이 한 달 소요되는 단위인 인원-월(MM: Man-Month)이며, 규모는 완성될 소프트웨어 시스템 규모의 크기를 나타내는 단위인 1,000라인, 즉 KDSI(Kilo-line Delivered Source Instruction)이고, c 와 k 는 상수이다. COCOMO 모델은 소프트웨어의 유형을 세 가지로 분류하여 단순형(organic), 중간형(semi-detached), 내장형(embedded)으로 나누어 c 와 k 를 추정하고 있다.

프로젝트 개발 유형별 특성은 표 1과 같다.

:: 표 1 | 개발 유형별 특성

개발 유형			프로젝트 특성	
형태	제품 크기	혁신성	일정 제한	개발 분야
단순형 (organic)	작음 (50 KDSI 이하)	적음	엄격하지 않음	상대적으로 단순한 소프트웨어 (예: 사무처리, 업무용, 일반 응용 프로그램 등)
중간형 (semi-detached)	보통 (50~300 KDSI)	보통	보통	크기와 복잡성 면에서 중간 정도의 소프트웨어 (예: 컴파일러, 인터프리터 등)
내장형 (embedded)	큼 (300 KDSI 이상)	많음	엄격함	규모가 크고 복잡한 소프트웨어 (예: 대형OS, DBMS, 미사일 유도 시스템 등)

프로젝트 개발 유형에 따라 적용되는 예상 노력 추정 공식은 표 2와 같다.

:: 표 2 | COCOMO 모델의 프로젝트 개발 유형별 예상 노력

프로젝트 개발 유형	공식(c & k)	유형 해설
단순형(organic)	$PM = 2.4 \times (KDSI)^{1.05}$	<ul style="list-style-type: none"> 프로젝트가 친숙하고 이전 프로젝트와 유사한 경우 안정적인 개발 환경인 경우 소프트웨어의 규모가 작은 경우
중간형(semi-detached)	$PM = 3.0 \times (KDSI)^{1.12}$	<ul style="list-style-type: none"> 단순형과 내장형의 중간형의 경우
내장형(embedded)	$PM = 3.6 \times (KDSI)^{1.20}$	<ul style="list-style-type: none"> 프로젝트 제약조건이 높고 일정이 촉박한 경우 크기가 크고 혁신적인 소프트웨어 개발 프로젝트의 경우 하드웨어와 결합된 실시간 시스템의 경우

공식에서 쉽게 알 수 있듯이 예상 노력은 원시 코드의 라인 수에 비례한다. 상수 c와 k는 모델마다 다르며, 지수 값 k는 1보다 크다. 그 이유는 프로젝트에 드는 비용이 규모가 커질수록 크기에 단순 비례하는 것보다 더 큰 노력이 들어간다는 것을 반영한 결과이다. 예를 들어, 10만 줄짜리 프로그램은 1만 줄짜리 프로그램에 비해 10배의 노력만 필요로 하는 것이 아니라 복잡도가 기하급수적으로 증가하여 실제 추가로 인력과 시간이 요구된다. 또한 프로젝트가 대형화될수록 프로젝트의 복잡도가 높아지고 제약 조건이 증가하여 프로젝트 관리를 위해 더 높은 기술력과 비용을 요구하게 된다.

예를 들어, 인터넷쇼핑몰 프로젝트는 단순형(organic) 5만 줄짜리 프로그램의 경우 COCOMO 모델을 적용하면 예상 노력 $E = 2.4 \times (50)^{1.05} = 146MM$ 이다.

이 모델은 일반적으로 널리 적용될 수 있는 장점이 있으나 정확도가 낮아 이를 보정하는 노력이 추가로 요구된다. 제품의 특성, 컴퓨터 속성, 개발 인력 특성, 프로젝트의 특성을 바탕으로 예측된 노력을 보정(scaling)해 주어야 한다. 이들 4가지 프로젝트의 특성을 총 15가지

비용 인자 항목을 사용하여 나누고, 각 항목의 추정치(보정 계수)를 곱하여 예측 노력 값을 구한다. 예를 들어, ‘제품의 특성’에 들어가는 비용 인자 항목으로는 ‘요구되는 신뢰도’, ‘데이터베이스 크기’, ‘제품의 복잡도’가 있다.

::표 3 | COCOMO 모델에서 사용되는 노력 보정 계수

특성 분류	비용 승수 요소	노력보정계수(Effort Adjustment Factor)					
		매우 낮음	낮음	정상	높음	매우 높음	극히 높음
제품의 특성	요구되는 신뢰도	0.75	0.88	1.00	1.15	1.40	
	데이터베이스 크기		0.94	1.00	1.08	1.16	
	제품의 복잡도	0.70	0.85	1.00	1.15	1.30	1.65
컴퓨터의 특성	실행 시간의 제약			1.00	1.11	1.30	1.66
	주기의 장치의 제약			1.00	1.06	1.21	1.56
	H/W, S/W의 안전성		0.87	1.00	1.15	1.30	
	처리 시간		0.87	1.00	1.07	1.15	
개발 요원의 특성	분석가의 능력	1.46	1.19	1.00	0.86	0.71	
	응용 경험	1.29	1.13	1.00	0.91	0.82	
	프로그래머 능력	1.42	1.17	1.00	0.86	0.70	
	컴퓨터와의 친숙성	1.21	1.10	1.00	0.90	-	
	프로그래밍 언어 경험	1.14	1.07	1.00	0.95		
프로젝트 성격	소프트웨어 공학 원리의 사용	1.24	1.10	1.00	0.91	0.82	
	소프트웨어 도구의 사용	1.24	1.10	1.00	0.91	0.83	
	요구되는 개발 일정	1.23	1.08	1.00	1.04	1.10	

예를 들어, 인터넷 쇼핑몰 프로젝트의 단순형 10만 줄짜리 프로그램의 경우 302MM의 예상 노력이 추정되었으며, 비용 인자를 분석해 보니 프로젝트의 특성상 요구되는 신뢰도가 높고(보정 계수 1.15), 실행 시간의 제약이 높으며(보정 계수 1.11), 프로그래머 능력이 매우 낮고(보정 계수 1.42), 소프트웨어 공학 원리의 사용이 매우 낮으며(보정 계수 1.24), 나머지 보정 계수가 정상(보정 계수 1.00)이라고 평가되었다고 가정하면,

$$\text{예상 노력 } E = (1.15 \times 1.11 \times 1.42 \times 1.24) \times 2.4 \times (50)^{1.05} = 327\text{MM이다.}$$

COCOMO 모델은 프로젝트를 완성하는 총 개발 기간을 추정하는 식을 제공한다. 추정된 일정과 프로젝트 계획에서 요구되는 일정이 반드시 똑같을 필요는 없다. 프로젝트의 총 개발

기간(TDEV: Total Development Time)은 예상 노력을 이용하여 다음과 같은 식으로 구한다.

- 단순형(organic) 프로젝트: $TEDV = 2.5 \times (E)^{0.38}$
- 중간형(semi-detached) 프로젝트: $TEDV = 2.5 \times (E)^{0.35}$
- 내장형(embedded) 프로젝트: $TEDV = 2.5 \times (E)^{0.32}$

위의 식을 보면 개발 기간은 프로젝트 개발 유형에 관계없이 비슷하다. 또한 소프트웨어 프로젝트의 규모가 커지더라도 총 개발 기간은 상대적으로 큰 변화가 없다. 이는 프로젝트 규모가 커지더라도 요구되는 활동들을 병렬 처리함으로써 개발 기간을 단축할 수 있음을 의미한다. 앞에서 예를 들은 인터넷 쇼핑몰 프로젝트의 경우 $TEDV = 2.5 \times (327)^{0.38} = 22.5$, 즉 22개월 정도의 총 개발 기간이 소요된다. 프로젝트 규모가 2배로 커진다고 가정하면 $TEDV = 2.5 \times (654)^{0.38} = 29.3$, 즉 29개월 정도의 총 개발 기간이 소요된다는 것을 알 수 있다.

LOC를 계산할 때 빈 줄이나 주석(코멘트)은 포함하지 않은 경우 논리적 LOC라 부른다. 주석이나 빈 줄을 포함한 물리적인 줄 수(키 보드의 'enter' 키가 눌린 횟수)를 나타낸 것을 물리적 LOC라 한다. LOC는 단순하고 적용이 쉬워 널리 사용되는 척도이나 정확도가 떨어진다는 단점이 있다. 코드의 행 수를 기반으로 소프트웨어 규모를 측정하는 것은 다음과 같이 여러 가지 이유에서 한계를 드러낸다[Jones 1996].

- 소프트웨어 규모의 크기는 고객 요구사항 크기로 반영되어야 하나 LOC는 고객의 요구사항 크기를 반영한 것이라 보기 어렵다.
- 경제적 의미가 미흡하고 언어 수준에 따라 결과가 왜곡되기도 한다.
- 개발 공수의 50% 이상이 투입되는 공정(분석, 설계, 문서화, 관리, 품질 등)은 코드의 행 수로 측정할 수 없다.
- 현재 사용되고 있는 언어만도 500개가 넘어, 이들 언어 간 생산성을 비교할 표준이 없다.
- 프로그램이 아직 작성되지 않은 상태에서 정확한 라인 수를 미리 예측하는 것이 불가능하다.
- 절차식이 아닌 풀다운(pull-down) 메뉴나 버튼 컨트롤 방식을 사용하는 객체 지향형 언어에서는 코드의 행 수를 사용하기가 곤란하다.
- 개발자의 역량에 따라 코드의 행 수가 달라질 수 있어 객관적인 규모 측정이 어렵다.

결국 LOC 추정 기법은 소프트웨어 개발 프로세스 초기에 항상 적용될 수 없다는 한계와 소프트웨어 수명주기에 걸쳐서 항상 동일하게 적용될 수 없다는 제약 조건을 가지고 있다. 또한 개발자에게 라인 수에 대한 생산성의 의미를 부여할 수 있지만 소프트웨어 사용자 또는

고객에게 개발자의 노력에 대한 의미 있는 해석을 항상 해줄 수는 없다.

COCOMO 모델은 소프트웨어 비용 측정에 사용되는 프로그래밍 언어의 영향을 받는다는 한계를 가지고 있다. COCOMO 모델은 복잡도가 다른 두 언어를 사용하여 생산성을 비교할 때 일관성을 유지하기 어려워 모순적인 결과를 초래할 수 있다. 고급 언어로 프로그래밍할 때 어셈블리를 사용하는 것보다 더 효율적이고 생산적이라는 것은 잘 알려져 있다. 만약 하나의 소프트웨어가 각각 어셈블리와 C++를 사용하여 작성되었다고 가정하자. 프로그래머의 어셈블리 코드 생산성은 월 700라인/월이고 C++는 300라인/월 정도이다. 그러나 같은 애플리케이션을 만들 때 어셈블리 버전은 더 많은 양의 라인 수를 필요로 하고, 늦게 인도되며 더 많은 개발 비용이 드는 것이 일반적이다. 만약 두 버전에 같은 비용이 소요된다고 가정하면 어셈블리가 더 생산적이라는 모순적 결과가 나타날 수 있다.

2 기능 점수 기반 비용 측정

고객의 요구를 만족시키기 위해 고객의 기대치를 계량화할 필요가 있다. 고객의 기대치는 소프트웨어의 기능 규모와 품질 수준으로 계량화하여 나타내고, 이를 다시 요구되는 노력과 기간으로 환산하여 프로젝트 스케줄로 표현할 수 있어야 한다. 제한된 예산과 일정 내에서 고객의 기대치에 부응하기 위한 체계적, 반복적, 가시적 노력이 소프트웨어 개발자의 궁극적인 과제라 할 수 있다. 기능 점수(function point)는 측정의 초점을 ‘소프트웨어가 어떻게 구현되었는지’에서 ‘사용자가 어떠한 기능을 요구했는지’로 이동시킴으로써 LOC의 제약 사항을 극복할 수 있도록 설계한 것이다

기능 점수는 소프트웨어가 가지는 기능의 개수를 기준으로 소프트웨어의 규모를 정량화한 것이다. 기능 점수 모델은 LOC 기반 공수 추정의 단점을 극복하기 위해 기능 점수에 기초하여 공수 측정하는 방법이다. 기능 점수는 소프트웨어 시스템의 규모를 사용자 관점에서 객관적으로 측정할 수 있는 척도를 제공한다. 기능 점수 분석은 1970년대에 IBM의 알브레트(Allan J. Albrecht)에 의해 처음 개발되었다. 이 분석 방법은 소프트웨어 크기의 척도로 쓰였던 LOC의 한계를 극복하고 소프트웨어 개발과 관련된 비용을 예측하는 메커니즘을 개발하기 위한 시도의 일환으로 만들어졌다.

기능 점수 분석 방법은 개발 언어나, 개발자의 개인 역량, 개발 공정에 따른 측정 대상의 다양성과 같은 요소를 배제하고, 사용자의 기능 요구사항에 기반하기 때문에 코드 행 수를 분석하는 방법보다 상대적으로 객관적인 규모 측정이 가능하다는 평가를 받는다. 기능 점수 분석 방법은 1986년 국제기능사용자그룹(IFPUG: International Function Point Users Group)이 설

립되기까지 정교화되었으며, 현재 IFPUG는 기능 점수 분석과 관련된 실무 매뉴얼을 여러 버전에 걸쳐서 출판하고 있다.

기능 점수 분석은 앞서 설명한 것처럼 사용자 관점에서 객관적인 수치를 이용해 소프트웨어 시스템의 규모를 나타낸다. 사용자란 기능적인 관점에서 시스템을 이해하는 사람들이다. 기능 점수 모델은 시스템을 더 잘 이해하고 분석할 수 있도록 보다 작은 부분으로 나누는 방법이다. 기능 점수 분석은 프로그래밍 언어와 상관없이 일정하며 개발 기술 및 품질 수준과는 독립적이며 측정이 매우 간단하고 일관성을 가지고 있다. 기능 점수 기법의 주요 단점은 정확한 측정을 위해 인증받은 기능 점수 전문가가 필요하며, 측정에 비용과 시간이 많이 소요되고, 비용을 추정해야 하는 프로젝트 초기 단계에 모든 기능을 다 파악하기 어렵다는 점이다.

소프트웨어 프로젝트를 시작하는 시점에서 그나마 제대로 파악할 수 있는 것은 시스템의 주요 기능 정도이다. 프로젝트 초기 단계에서 사용자의 요구사항은 급변한다. 프로젝트 초기 단계에서 정확한 기능 점수 측정은 불가능하며, 알려지지 않은 기능과 그 기능의 복잡도를 가정하여 추정해야 한다. 기능 점수 모델은 개발 라이프 사이클 전 과정에서 반복적으로 활용될 수 있다는 장점이 있으며, 프로젝트가 진행되며 정확한 기능 점수 측정이 가능하다.

기능 점수 분석은 소프트웨어 시스템을 보다 작은 구성 요소로 분할해 나가면서 측정하는 하향식 접근 방법을 적용하고 있다. 기능 점수를 분석하는 데 사용되는 방법은 해당 분야나 상황에 따라 다양하지만, 이곳에서는 알브레트에 의해 개발되고 IFPUG에 의해 개정된 규칙을 기반으로 설명하고자 한다.

기능 점수 항목

소프트웨어의 규모와 복잡도는 사용자에게 제공하는 트랜잭션과 이를 지원하는 데이터에서 찾을 수 있다. 기능 점수 추정은 요구사항 분석으로부터 시작하며 사용자에게 제공되는 소프트웨어의 기능을 다음 5가지 핵심 항목들로 나누고, 각 항목에 가중치를 부여한 후, 주어진 가중치의 합으로 구성한다. 가중치는 경험과 프로젝트의 특성에 따라 달리 부여될 수 있다.

- 외부 입력(EI: External Input): 애플리케이션 밖에서, 즉 사용자 또는 다른 응용 프로그램으로부터 애플리케이션 안으로 자료를 가져오는 기본 프로세스 또는 트랜잭션
- 외부 출력(EO: External Output): 애플리케이션 내에서 애플리케이션 경계 밖으로 자료를 취해 가는 프로세스 또는 트랜잭션

- 외부 조회(EQ: External Inquiry): 애플리케이션의 응답을 온라인 출력의 형태로 내보내게 하는 온라인 입력
- 내부 논리 파일(ILF: Internal Logical File): 애플리케이션 내에 존재하는 데이터를 논리적으로 모아 놓은 것
- 외부 인터페이스 파일(EIF: External Interface File): 애플리케이션 외부에 있는 데이터이며 애플리케이션에서 사용할 수 있는 정보 제공

가중치의 값은 유사 프로젝트 경험과 프로젝트 성격에 따라 달라질 수 있다.

::표 4 | 기능점수 항목과 가중치의 예

기능 항목	예	가중치 인자(weighting factor)		
		단순	보통	복잡
외부 입력	화면, 신호	3	4	6
외부 출력	보고서, 화면, 에러 메시지	4	5	7
외부 조회	데이터 검색	3	4	6
내부 논리 파일	어플리케이션 경계 내 저장 데이터	7	10	15
외부 인터페이스 파일	외부 애플리케이션 데이터	5	7	10

각 기능 항목의 개수와 복잡도를 나타내는 가중치 인자가 결정되면 이를 곱하여 각 기능 항목의 기능 점수를 구한 후 이를 합하면 애플리케이션의 기능 점수가 계산된다. 예를 들어, 각 기능 항목의 개수와 가중치 인자가 결정되었다면 기능 점수는 다음과 같이 계산된다.

::표 5 | 복합 가중치 인자를 이용하여 기능 점수를 구하는 예

기능 항목	개수	단순	보통	복잡	FP=개수×가중값
입력(트랜잭션)	7	3	④	6	28
출력(화면 및 출력 양식)	6	4	5	⑦	42
질의	4	③	4	6	12
파일	8	7	⑩	15	80
응용 인터페이스	③	5	7	⑩	30
계					192

각 기능 항목의 개수와 가중치를 곱한 값을 합한 기능 점수를 미조정 기능 점수(UFC: Unadjusted Function-point Count)라 한다. 미조정 기능 점수는 규모 보정 계수, 애플리케이션

유형 보정 계수, 언어 보정 계수, 품질 및 특성 보정 계수 등 시스템의 특성 및 복잡도를 반영한 조정 인자들에 의해 보정되어 조정 기능 점수로 산정된다.

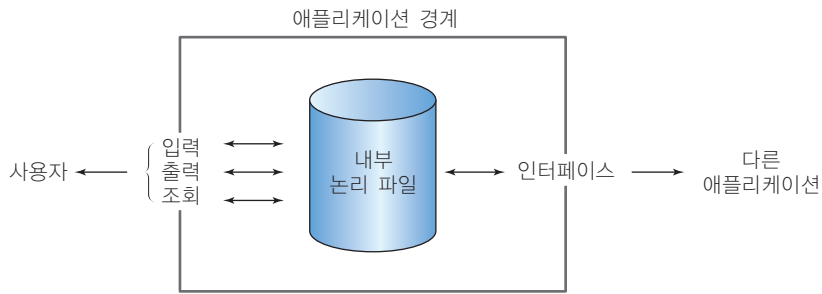
기능 점수 산정 절차

기능 점수는 크게 데이터 기능과 트랜잭션 기능으로 구분된다. 데이터 기능에서 데이터란 측정 대상 애플리케이션 내부에 저장되는 데이터와 측정 대상 애플리케이션이 외부 애플리케이션을 통해 참조하는 데이터를 의미한다. 이들 데이터를 논리 데이터라 부르며, 흔히 여러 속성들로 구성된 엔티티 또는 클래스들로 표현된다. 여기서 사용자가 사용하는 정보의 구조에 초점을 맞추고, 데이터가 애플리케이션에 어떻게 유지되고 저장되는지에 관한 세부 방법은 중요하지 않다.

트랜잭션 기능에서 트랜잭션이란 측정 대상 애플리케이션을 통해 사용자가 수행할 수 있는 단위 프로세스이다. 사용자는 애플리케이션을 통해서 데이터를 입력, 수정, 삭제하거나 조회할 수 있으며, 특별히 가공된 데이터를 요청할 수도 있다. 이러한 단위 프로세스들을 기능 점수에서는 트랜잭션 기능이라 부른다.

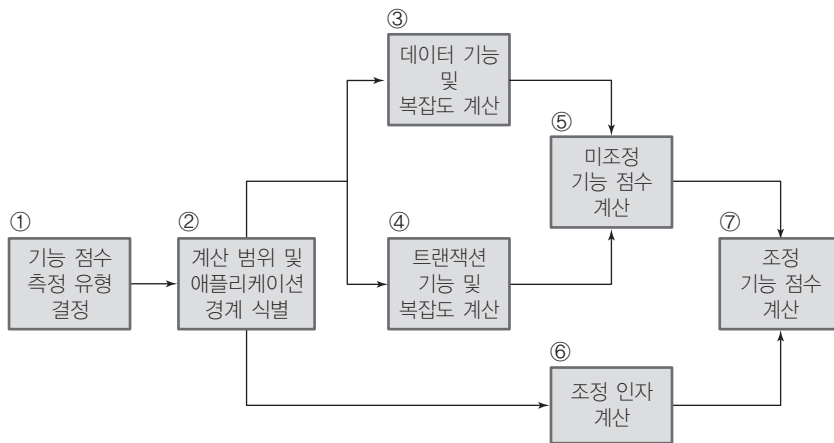
사용자가 애플리케이션을 통해 사용하게 되는 데이터와 단위 프로세스, 즉 트랜잭션의 수를 통해 측정 대상 애플리케이션의 규모를 측정하는 것이 기능 점수의 핵심이라 할 수 있다. 기능 점수를 산정하기 위해 고려해야 하는 요소들을 시스템 배경도의 형태로 표현하면 그림 1과 같다. 그림 1에서 내부 논리 파일(ILF: Internal Logical Files)은 측정 대상 애플리케이션에 저장되는 논리 데이터를 의미한다. 인터페이스는 외부 인터페이스 파일(EIF: External Interface File)로 측정 대상 애플리케이션이 외부 애플리케이션으로부터 데이터를 얻어오기 위해 존재하는 것이다.

트랜잭션 기능은 애플리케이션이 사용자에게 제공하는 서비스를 나타낸다. 트랜잭션 기능은 사용자(users)가 측정 대상 애플리케이션을 통해 수행하는 단위 프로세스이며, 외부 입력(EI: External Inputs), 외부 출력(EO: External Outputs), 외부 조회(EQ: External inQuires)로 구분한다. 입력은 말 그대로 사용자가 어떤 데이터를 애플리케이션 내부에 유지시키는 행위로서, 여기에는 추가, 수정, 삭제와 같은 처리가 포함된다. 출력은 사용자가 애플리케이션을 통해 어떤 정보를 확인하고자 할 때, 저장된 데이터들을 가공 처리하여 보여주는 것을 의미한다. 조회는 사용자가 애플리케이션 내부에 저장된 특정 논리 데이터를 있는 그대로 보여주는 경우이다.



::그림 1 | 기능 점수의 측정 요소

데이터 기능과 트랜잭션 기능을 이용한 구체적인 기능 점수 산정 방법은 조금 뒤에 설명하도록 하고, 우선 기능 점수의 일반적인 산정 절차에 대해 설명하고자 한다[한국전산원 2001]. 그림 2는 IFPUG에서 정의한 기능 점수 산정 절차이다.



::그림 2 | 기능 점수 산정 절차 (CPM 4.1, IFPUG, 1999)

(1) 기능 점수 측정 유형 결정

기능 점수 산정에서 가장 먼저 필요한 것은 측정 유형(신규 개발, 유지보수, 이미 개발된 패키지)을 결정하는 것이다. 개발되는 애플리케이션이 어떠한 시점과 목적에 의해 만들어지는지를 고려하여 측정 유형을 결정한다. 측정 유형은 다음과 같이 크게 세 가지로 구분된다.

① 개발 프로젝트 기능 점수

개발 프로젝트 기능 점수(DFP: Development Project Function Point)는 일반적으로 신규 개발 프로젝트를 진행하려는 경우에 사용자에게 제공하는 기능을 측정하는 것이다. 개발 프로젝트 기능 점수 측정의 가장 일차적인 목표는 프로젝트를 진행하기 위해 필요한 예산, 인력, 시

간 등을 예측하기 위한 것이다. 다시 말해, 프로젝트가 본격적으로 진행되지 않은 상태, 즉 프로젝트 초기 단계에 고객의 기능 요구사항을 토대로 프로젝트 진행에 소요되는 자원들의 규모를 측정하고 그 대가를 산정하는 것이 목적이다.

개발 프로젝트 기능 점수는 프로젝트가 진행되는 동안 매 단계에서 측정될 수 있으며, 그 결과는 프로젝트 초기에 측정한 결과와 비교하는 검증 자료로 사용된다. 이들 비교를 통해 ‘범위 변경(scope creep)’에 따라 추가된 기능을 포착할 수 있다. 프로젝트가 모두 완료된 시점에서 측정된 기능 점수는 기준선 기능 점수라고도 하며, 그 결과는 향후 발생할 수도 있는 유지보수 작업 등의 규모를 측정하는 데 기준 정보로 사용된다. 이 기능 점수 유형은 SI(System Integration) 개발 프로젝트의 기능 점수를 측정하고자 하는 경우 적용될 수 있다. 참고로 국내 소프트웨어 사업 대가 기준에서는 개발 프로젝트 기능 점수만을 인정한다.

② 개선 프로젝트 기능 점수

개선 프로젝트 기능 점수(EFP: Enhancement Project Function Point)는 기존 애플리케이션에 대한 변경을 측정하는 것으로 유지보수 프로젝트의 개선 요구사항이 밝혀진 시점에 계산된다. 현재 사용 중인 애플리케이션에 새롭게 추가되는 기능, 삭제되는 기능, 변경되는 기능에 대해서 측정한다. 이 유형은 유지보수 프로젝트를 진행하는 경우, 유지보수에 필요한 인력이나 비용을 산정할 때 적용될 수 있다.

③ 애플리케이션 기능 점수

애플리케이션 기능 점수(AFP: Application Function Point)는 현재 운용 중인 애플리케이션에 대한 기능을 측정하는 것이다. 설치된 애플리케이션의 기능 점수를 측정하는 이유는 향후에 또 다른 개선 요구사항이 발생하는 경우, 애플리케이션의 개선 작업을 토털 아웃소싱 형태로 계약하여 진행할 경우, 운영 중인 애플리케이션에 대한 기능 점수를 토대로 개선 작업에 대한 규모를 측정해야 하기 때문이다. 이를 위해 현재 애플리케이션의 기능 규모와 추가, 수정, 삭제될 기능을 측정하여 비용을 산정하고 계약을 맺어야 한다. 애플리케이션 기능 점수는 이런 경우에 먼저 측정하게 되며, 향후 개선 요구사항이 모두 완료된 후에도 다시 측정할 수 있다. 이러한 이유로 애플리케이션 기능 점수를 기준선(baseline), 또는 설치된(installed) 기능 점수라 부르기도 한다.

(2) 계산 범위 및 애플리케이션 경계 식별

기능 점수 산정에서 두 번째 절차는 측정 범위와 경계를 식별하는 것이다. 측정 범위란 규모 측정 대상 소프트웨어의 특징 및 기능의 집합을 의미하며, 사용자의 기능적 요구사항을

식별하여 정의하고 문서화하는 것에서 출발한다. 경계(boundary)는 측정 대상 애플리케이션과 외부 사이의 경계선을 말한다.

경계 식별은 시스템에 포함시킬 요구사항과 제외시킬 요구사항을 정의하며, 사용자(또는 외부 애플리케이션)와 측정 대상 애플리케이션이 상호 어떤 데이터를 주고받으며 어떠한 상호 교류가 일어나는지 밝히는 것이다. 경계는 내부 애플리케이션과 외부 사용자 세계 간의 개념적인 인터페이스이며, 트랜잭션(EI, EO, EQ)에 의해 처리된 데이터가 애플리케이션에 들어 나가는 출입문과 같은 역할을 한다. 애플리케이션의 경계를 확실하게 구분해야 데이터 기능이나 트랜잭션 기능을 정확하게 계산해 낼 수 있다. 애플리케이션의 범위를 식별한 결과는 사용자의 기능적 요구사항을 나타내는 유스케이스 모델 등으로 표현될 수 있다. 애플리케이션의 경계는 그림 1과 같은 배경도의 형태로 표현된다.

(3) 데이터 기능 및 복잡도 계산

데이터 기능의 계산은 사용자 요구사항이나 논리 데이터 모델(예를 들어, ER 모델, 클래스 다이어그램 등), 프로세스 모델(DFD: 자료 흐름도), 화면 등을 토대로 데이터 그룹을 도출하는 것에서부터 시작한다. 데이터 그룹에서 데이터가 측정 대상 애플리케이션 경계 내부에 유지되는가 아니면 외부 애플리케이션에서 유지되는가를 판단하여 내부 논리 파일(ILF)과 외부 인터페이스 파일(EIF)로 구분되었다. 이들은 모두 논리적으로 관련된 데이터나 제어 정보의 그룹으로서 사용자가 식별 가능한 것이어야 한다. ILF는 측정 대상 애플리케이션의 경계 내부에서 저장된다. EIF는 다른 애플리케이션 내에 저장되고, 측정 대상 애플리케이션에서 참조된다. ILF와 EIF를 구분한 후에는 각 논리 파일들에 대한 복잡도를 분석하여 최종 데이터 기능 점수를 계산한다. 복잡도는 각 논리 파일이 다른 파일과 어떠한 연관 관계를 맺고 있는지, 각 파일들을 구성하는 속성들이 얼마나 많은지에 따라서 결정된다.

(4) 트랜잭션 기능 및 복잡도 계산

트랜잭션 기능의 계산은 단위 프로세스(elementary process)를 식별하는 것에서부터 시작한다. 단위 프로세스란 사용자가 대상 애플리케이션에 대해 수행하는 작업을 의미하며, 식별된 단위 프로세스는 유형에 따라서 세 가지(외부 입력, 외부 출력, 외부 조회)로 구분된다. 이 작업들은 애플리케이션 외부에서 사용자가 수행하는 기능이다. 사용자는 애플리케이션 밖에서 애플리케이션이 제공하는 기능을 사용하기 때문에 각 트랜잭션의 기능에 ‘외부’라는 표현이 붙는다.

세 가지 유형의 트랜잭션 기능이 식별되면 각각에 대해서 복잡도를 분석한다. 복잡도는 각 트랜잭션 기능에서 사용하는 데이터(ILF 또는 EIF)의 수와 이들 데이터에 속한 속성들의 수에

따라 결정된다.

(5) 미조정 기능 점수 계산

데이터 기능과 트랜잭션 기능에 대한 계산이 끝나면 두 기능에 대한 점수를 합산하여 미조정 기능 점수(UFP: Unadjusted Function Point)를 구한다. 미조정 기능 점수는 애플리케이션에 고려되어야 하는 여러 가지 특성들(예를 들어, 애플리케이션 유형, 언어, 품질)을 반영하지 않고 단순히 기능적인 요구사항만을 고려한 것이다. 다음 단계에서 미조정 기능 점수는 계산된 조정 인자와 조합되어 최종 기능 점수로 산출된다.

(6) 조정 인자 계산

정보 시스템이 제공하는 기능에는 데이터 기능과 트랜잭션 기능에 의해 충분히 표현되지 않는 특성들이 있다. 조정 인자(VAF: Value Adjustment Factor)는 이러한 특성들이 측정 대상 애플리케이션에 미치는 영향 정도를 파악한 후, 이를 일정한 수식에 대입하여 계산하는 수치로서, 조정 기능 점수(AFP: Adjusted Function Point)의 계산을 위한 승수(multiplier)로 사용된다.

국내의 경우에는 기능 점수 산정 시 지식경제부에서 고시한 소프트웨어 사업 대가 기준에 따라 다음과 같은 보정 계수를 조정 인자로 사용한다[지식경제부 2010].

- 규모 보정 계수
- 애플리케이션 유형 보정 계수
- 언어 보정 계수
- 품질 및 특성 보정 계수

데이터 기능과 트랜잭션 기능의 점수를 합산한 미조정 기능 점수에 위의 네 가지 보정 계수를 곱하여 최종 기능 점수, 즉 조정 기능 점수를 구한다. 다음은 위의 네 가지 보정 계수에 대한 설명이다.

① 규모 보정 계수

규모 보정 계수는 측정 대상 애플리케이션의 미조정 기능 점수의 규모가 얼마나 큰지 파악하여 그 규모만큼 조정 인자를 반영하는 것이다. 일반적으로 기능의 규모가 크면 클수록 애플리케이션의 복잡도가 더욱 증가하기 때문에 보정 계수 또한 높아지게 된다. 데이터 기능과 트랜잭션 기능을 합산한 결과가 300 이상인 경우의 보정 계수는 아래의 식을 통해 계산되며, 기능 점수가 300 미만인 경우에는 0.65라는 일정한 보정 계수를 적용한다.

$$\text{규모 보정 계수} = 0.108 \times \log_e(\text{기능 점수}) + 0.2229$$

② 애플리케이션 유형 보정 계수

애플리케이션 유형 보정 계수는 측정 대상 애플리케이션이 어떠한 목적으로 개발되는지에 따라 정해진다. 애플리케이션 유형 보정 계수와 애플리케이션의 유형을 분류하는 기준은 표 6에 나타나 있다.

::표 6 | 애플리케이션 유형 보정 계수

애플리케이션 유형	범위	보정 계수
업무처리용	인사, 회계, 급여, 영업 등 경영 관리 및 업무처리용 소프트웨어 등	1.0
과학기술용	과학계산, 시뮬레이션, 스프레드시트, 통계, OR, CAE 등	1.2
멀티미디어용	그래픽, 영상, 음성 등 멀티미디어 응용 분야, 지리정보시스템, 교육/오락용 등	1.3
지능정보용	자연어 처리, 인공지능, 전문가 시스템	1.7
시스템용	운영체제, 언어처리 프로그램, DBMS, 인간/기계 인터페이스, 윈도우 시스템, CASE, 유틸리티 등	1.7
통신제어용	통신 프로토콜, 에뮬레이션, 교환기 소프트웨어, GPS 등	1.9
공정제어용	생산관리, CAM, CIM, 기기제어, 로봇제어, 실시간, 내장형 소프트웨어 등	2.0
지휘통제용	군, 경찰 등 군 장비, 인력의 지휘통제를 요하는 소프트웨어	2.2

출처: 지식경제부고시 제2010-52호, 소프트웨어사업 대가의 기준-제8조 관련

③ 언어 보정 계수

언어 보정 계수는 발주자(또는 고객)가 특정 언어를 요구하는 경우, 사용된 언어의 비율에 따라 소프트웨어 개발 단계 중 구현과 시험 단계에만 적용한다. 개발 언어에 따른 보정 계수는 표 7과 같다.

::표 7 | 언어 보정계수

언어 구분	보정 계수
Assembly, 기계어, 자연어	1.9
C, CHILL, C++, JAVA, C#, PROLOG, UNIX Shell Scripts	1.2
COBOL, FORTRAN, PL/1, PASCAL, Ada	1.0
ABAP4, Delphi, HTML, Power Builder, Program Generator, Query default, Small Talk, SQL, Visual Basic, Statistical default, XML default, Script default(JSP, ASP, PHP, Flash 등)	0.8
EXCEL, Spreadsheet, default, Screen painter default	0.6

출처: 지식경제부고시 제2010-52호, 소프트웨어사업 대가의 기준-제8조 관련

④ 품질 및 특성 보정 계수

마지막으로 품질 및 특성 보정 계수는 측정 대상 애플리케이션이 성능이나 신뢰성 등의 품질 요소에 의해 얼마나 영향을 받는지를 고려하여 결정한다. 품질과 관련된 요소마다 0~2 범위의 영향도를 정하고 있으며(총 8점), 각 요소의 영향도를 합산한 총 영향도를 이용하여 보정 계수를 산정한다. 품질 및 특성과 관련된 보정 요소 및 영향도 목록은 표 8에 나타나 있으며, 품질 및 특성 보정 계수를 구하기 위한 식은 아래와 같다. 만약 모든 영향도가 2인 경우, 총 영향도는 8이 되어 품질 및 특성 보정 계수의 값은 1.2가 된다.

$$\text{품질 및 특성 보정 계수} = 0.025 \times \text{총 영향도} + 1$$

(총 영향도 = 분산 처리 영향도 + 성능 영향도 + 신뢰성 영향도 + 다중 사이트 영향도)

::표 8 | 품질 및 특성 보정 요소 및 영향도

보정 요소		판단 기준	영향도
분산 처리	애플리케이션이 구성 요소 간에 데이터를 전송하는 정도	분산처리에 대한 요구사항이 명시되지 않음	0
		클라이언트/서버 및 웹 기반 애플리케이션과 같이 분산 처리와 자료 전송이 온라인으로 수행됨	1
		애플리케이션상의 처리 기능이 복수개의 서버 또는 프로세서상에서 동적으로 상호 수행됨	2
성능	응답시간 또는 처리율에 대한 사용자 요구 수준	성능에 대한 특별한 요구사항이나 활동이 명시되지 않으며, 기본적인 성능이 제공됨	0
		응답시간 또는 처리율이 피크타임 또는 모든 업무시간에 중요함. 연동 시스템의 처리 마감시간에 대한 제한이 있음	1
		성능 요구사항을 만족하기 위해 설계 단계에서부터 성능 분석이 요구되거나, 설계/개발/구현 단계에서 성능 분석 도구가 사용됨	2
신뢰성	장애 시 미치는 영향의 정도	신뢰성에 대한 요구사항이 명시되지 않으며, 기본적인 신뢰성이 제공됨	0
		고장 시 쉽게 복구 가능한 수준의 약간 불편한 손실이 발생함	1
		고장 시 복구가 어려우며, 재정적 손실이 많이 발생하거나, 인명 피해 위험이 있음	2
다중 사이트	상이한 하드웨어와 소프트웨어 환경을 지원하도록 개발되는 정도	설계 단계에서 하나 이상의 설치 사이트에 대한 요구사항만 고려됨. 애플리케이션이 동일한 하드웨어 또는 소프트웨어 환경하에서만 운영되도록 설계됨	0
		설계 단계에서 하나 이상의 설치 사이트에 대한 요구사항이 고려됨. 애플리케이션이 유사한 하드웨어 또는 소프트웨어 환경하에서만 운영되도록 설계됨	1
		설계 단계에서 하나 이상의 설치 사이트에 대한 요구사항이 고려됨. 애플리케이션이 상이한 하드웨어 및 소프트웨어 환경하에서 동작하도록 설계됨	2

출처: 지식경제부고시 제2010-52호, 소프트웨어사업 대가의 기준-제8조 관련

(7) 조정 기능 점수 계산

앞의 네 가지 보정 계수(규모 보정 계수, 애플리케이션 유형 보정 계수, 언어 보정 계수, 품질 특성 보정 계수)를 구하고 나면, 전 단계에서 계산한 미조정 기능 점수와 네 가지 보정 계수를 곱하여 최종 기능 점수, 즉 조정 기능 점수를 계산한다.

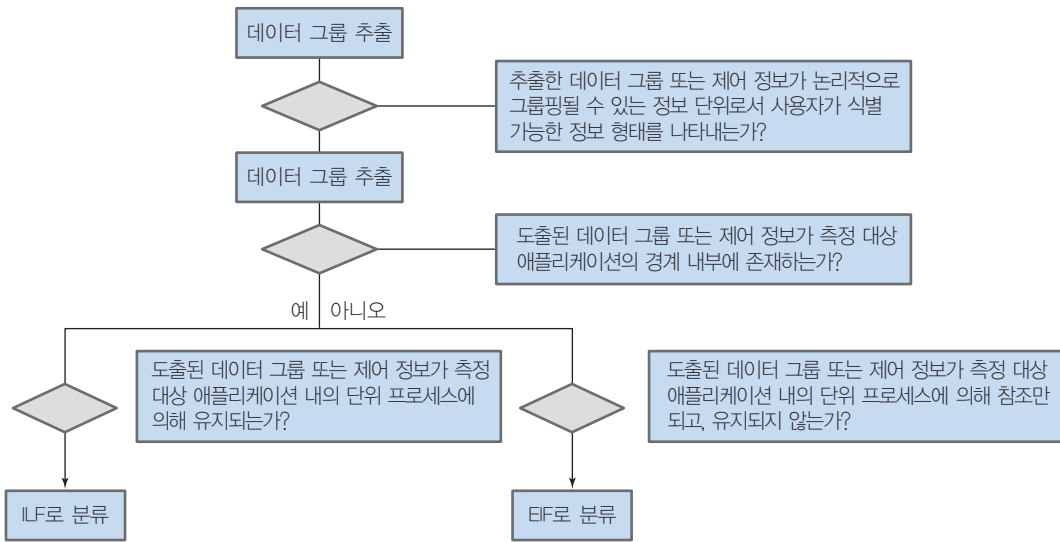
지금까지 기능 점수를 산정하는 전반적인 절차에 대해서 설명하였다. 앞서 설명했듯이 기능 점수의 산정은 앞으로 개발될 시스템, 이미 개발된 시스템, 개선되어야 할 시스템에 대해서 규모를 측정하기 위해 사용된다. 이미 개발되었거나 개선되어야 할 시스템의 경우, 기능 점수 산정에 필요한 정보들이 이미 존재할 수 있으나, 개발될 시스템의 경우 개발 초기 단계에서 활용할 수 있는 정보의 양이 많지 않다. 하지만 개발이 진행됨에 따라 기능 점수 산정에 유용한 정보를 줄 수 있는 문서의 종류도 다양해질 수 있다. 이러한 문서들에는 Use Case 모델, ER(Entity Relation) 다이어그램, 데이터 흐름도와 같은 것들이 포함된다.

다음 절에서는 기능 점수 산정의 핵심인 데이터 기능과 트랜잭션 기능의 측정방법에 대해서 보다 자세하게 설명하고자 한다.

데이터 기능 점수 측정

데이터 기능은 애플리케이션에 저장된 논리 데이터와 관련이 있으며 갱신, 참조, 검색을 위해 활용된다. 데이터의 복잡도나 데이터 기능 점수를 파악하기 위해 시스템의 요구되는 정보를 분석한 ER 다이어그램 또는 클래스 다이어그램이 활용될 수 있다. 현실적으로 보면 프로젝트 계획 단계에서 기능 점수를 추정할 때 이러한 다이어그램을 가지고 있지 않은 경우가 대부분이다. 따라서 위와 같은 결과물은 분석이 완료된 시점에 활용될 수 있으며, 프로젝트 초기 단계에서는 요구된 기능(필요하면 유스케이스 활용)에서 필요로 하는 논리 데이터의 구조를 예상하여 측정하며 복잡도는 사업 대가 기준에서 제시한 평균 복잡도를 적용하는 경우가 일반적이다.

내부 논리 데이터인 ILF는 측정 대상 애플리케이션 내에 유지(추가, 수정, 삭제 등)되며 사용자가 식별 가능하고 논리적으로 연결된 것이다. ILF는 세 가지 트랜잭션 기능(외부 입력, 외부 출력, 외부 조회)에 의해서 읽히거나 참조될 수 있다. 외부 인터페이스 데이터인 EIF는 측정 대상 애플리케이션 경계 내에서 참조되지만 다른 애플리케이션에서 유지되는 정보이다. EIF는 다른 애플리케이션에 저장되어 있다 하더라도 사용자가 식별 가능하고 측정 대상 애플리케이션 내에서 최소한 하나의 외부 입력, 외부 출력, 외부 조회에 의해서 참조될 수 있다. ILF와 EIF를 식별하는 규칙은 그림 3과 같이 나타낼 수 있다.



::그림 3 | ILF와 EIF 식별 규칙

출처: 지식경제부고시 제2010-52호, 소프트웨어사업 대가의 기준

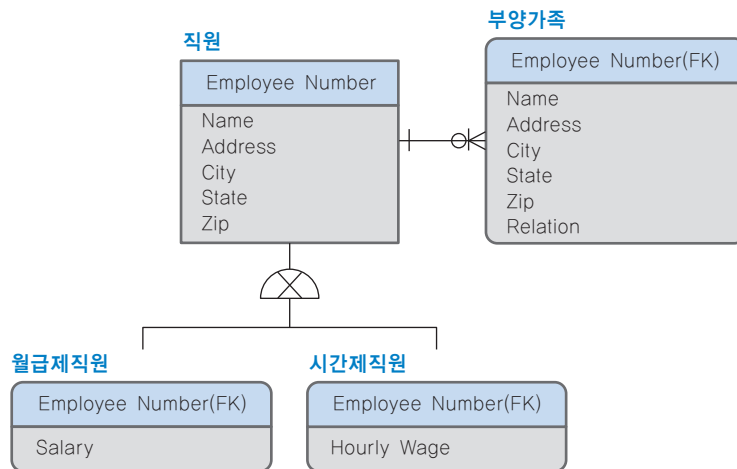
그림 3과 같은 절차를 통해 ILF와 EIF가 모두 식별되면, 각각에 대해서 복잡도를 계산해야 한다. 각 ILF와 EIF에 대한 복잡도는 구체적으로는 엔티티 타입, 관계 타입, 속성의 수에 의해 결정된다. 애플리케이션의 정보에 대한 분석이 이루어지기 전 데이터에 대한 추정이 이루어질 가능성이 높아 배경도 또는 유스케이스 시나리오를 활용하게 된다. 각 ILF와 EIF의 복잡도는 일반적으로 레코드 요소 유형(RET: Record Element Type)과 데이터 요소 유형(DET: Data Element Type)의 수를 기준으로 결정된다.

데이터 요소 유형은 ILF 또는 EIF에서 외래 키(foreign key) 속성을 포함하여 파일을 구성하는 각각의 필드(또는 속성)이다. 레코드 요소 유형은 ILF나 EIF에서 사용자가 식별 가능한 데이터 요소의 서브그룹이다. 서브그룹이란 이러한 필드들이 그룹화된 것으로 한 파일(ILF 또는 EIF)의 부분 집합 정도로 볼 수 있다. ER 다이어그램의 엔티티 타입, 관계 타입 또는 이들의 묶음이 RET가 될 수 있다. 하나의 레코드 요소에는 데이터 요소의 서브그룹(예를 들어, 엔티티 타입 또는 관계 타입)이 적어도 한 개 이상 존재한다.

RET는 선택적 서브그룹과 필수적 서브그룹 두 가지 유형으로 구분된다. 선택적 서브그룹은 사용자가 데이터의 인스턴스를 추가하고자 할 때, 입력할 수도 있고 그러지 않을 수도 있는 데이터 요소 그룹이다. 사용자는 적어도 하나 이상의 필수적 서브그룹을 사용해야 한다. 예를 들어, 직원 정보를 저장하고자 할 때, 직원은 월급제 또는 시간제 중 하나에 필수적으로 속해야 한다. 부양가족 정보는 입력해도 되고 하지 않아도 되는 선택적 서브그룹에 해당된다.

다. 또한 직원 정보는 기본적인 사항 외에도 월급제 또는 시간제 정보를 포함할 수 있다. 이 경우, 두 직원 타입은 다른 속성들을 갖게 되어 두 개의 RET로 규명된다. 또한 직원 정보에는 부양가족에 대한 정보가 포함될 수 있다. 이 정보는 선택적인 입력 정보로서 독립적으로 관리되어야 하므로 한 개의 RET로 식별된다.

지금까지 설명한 직원 정보는 그림 4와 같이 표현될 수 있다. 그림 4에서 월급제직원과 시간제직원은 직원과 상속 관계(일반화 관계)에 있으며, 기본적인 직원 정보는 직원 엔티티 타입에 포함되어 있다. 직원 엔티티 타입과 부양가족 엔티티 타입은 일대다 관계로서 직원과 부양가족과 선택적으로 관계를 맺는다. 다시 말해, 직원에게 속한 부양가족이 없을 수도 있고 여럿 있을 수도 있다. RET는 월급제직원, 시간제직원, 부양가족 3가지로 구별될 수 있다. 만약 그림 4와 같은 ER 다이어그램이 이미 존재한다고 가정하면 RET의 개수를 보다 쉽게 판단할 수 있다. 즉, 직원 엔티티 타입은 월급제직원과 시간제직원 둘 중 하나로 구체화될 수 있다는 점에서 2개의 RET가 규명되며, 부양가족 엔티티 타입은 직원 엔티티 타입으로 인해 유도되는 약 엔티티 타입(weak entity type)으로서 선택적인 서브그룹으로 판단되어 별개의 RET로 규명될 수 있다.



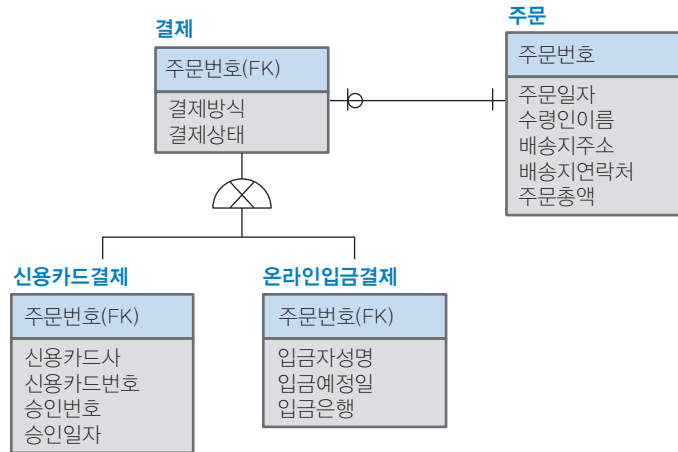
:: 그림 4 | 직원 정보를 나타내는 ER 다이어그램

- 월급제직원(필수적): 기본 사항 정보 포함
- 시간제직원(필수적): 기본 사항 정보 포함
- 부양가족(선택적)

그림 5는 인터넷 쇼핑몰의 주문-결제 정보와 관련된 ER 다이어그램을 보여주고 있다. 주문 엔티티 타입과 결제 엔티티 타입은 일대일 관계를 맺고 있다. 주문은 결제가 이루어지지

않은 경우도 있을 수 있어 선택적인 관계로 결제와 연결된다. 신용카드결제와 온라인입금결제는 결제 엔티티 타입과 상속 관계를 맺고 있으며, 이들 두 엔티티 타입은 결제 방식을 나타내기 위한 엔티티 타입이다. 주문-결제 정보에서 주문 엔티티 타입은 사용자가 식별할 수 있는 논리 데이터로서 하나의 ILF로 식별된다. 주문 엔티티 타입을 구성하는 데이터 요소(속성)들은 서브그룹으로 나눌 수 없기 때문에 하나의 RET로 계산한다. 주문 엔티티 타입과 관계를 맺고 있는 결제 엔티티 타입은 주문 엔티티 타입 없이 독립적으로 존재할 수 없는 약 엔티티 타입이다. 그림 4의 부양가족 엔티티 타입과 같이 결제 엔티티 타입 또한 독립적인 ILF로 보기보다는 주문-결제 정보를 나타내는 데이터 요소의 서브그룹으로 보는 것이 좋다. 결제 정보는 결제방식(신용카드결제, 온라인입금결제)에 따라서 두 가지 유형으로 구분될 수 있다. 결과적으로 주문-결제 정보는 아래와 같이 3개의 RET로 계산될 수 있다.

- 주문(필수적)
- 신용카드결제(선택적)
- 온라인입금결제(선택적)



::그림 5 | 주문-결제 정보를 나타내는 ER 다이어그램

ILF나 EIF에 대해서 RET를 계산하는 규칙을 정리하면 다음과 같다.

- ILF나 EIF의 선택적인 서브그룹이나 필수적인 서브그룹 각각을 하나의 RET로 계산한다.
- 만약 상속(parent-child) 관계를 나타내는 서브그룹이 존재하지 않으면, ILF나 EIF를 하나의 RET로 계산한다.

데이터 요소 유형(DET)은 ILF나 EIF에서 유지되거나 참조되는 필드로, 사용자가 식별할 수 있는 유일한 속성(필드) 각각에 대해서 하나의 DET로 계산한다. 예를 들어, 직원 정보를 구성하는 직원번호, 직원이름, 직무 등은 각각 유일한 필드로서 각각 하나의 DET로 계산한다. 참고로 다른 ILF 또는 EIF와의 관계를 설정하기 위해 필요로 하는 외래 키의 경우 하나의 DET로 계산한다. 또한 직원번호는 직원 레코드의 주 키로 사용되며, 이 필드는 직원의 부양가족을 나타내는 레코드에서 외래 키로 사용될 수 있다. 이 경우, 직원번호는 두 번 나타나지만 DET는 1로 계산한다. DET를 계산하기 위한 추가적인 규칙은 다음과 같다[한국전산원 2001].

- 기술이나 구현상의 이유 때문에 ILF나 EIF 내에서 여러 번 나타나는 필드는 오직 한 번만 계산한다.
- 포맷이 동일한 반복 필드는 ILF나 EIF 내에서 오직 한 번만 계산한다.
예) 12개월 동안 이루어진 각 월간 판매 실적 필드와 이를 모두 더한 하나의 연간 합계 판매 실적 필드는 두 개의 DET로 간주한다.
- 이벤트가 발생한 시간을 기록하는 타임 스탬프는 하나의 DET로 계산한다.
- 외부 입력(EI)을 처리하는 동안 내부에서 처리되어 데이터베이스에 저장되는 계산은 하나의 DET로 간주한다.

ILF와 EIF에 대해서 RET 및 DET의 계산이 모두 완료되면 이를 바탕으로 복잡도를 분석하여 최종 기능 점수를 산정한다. 국내 소프트웨어 사업 대가 기준에서는 데이터 기능의 복잡도를 결정할 수 있도록 다음과 같은 복잡도 매트릭스를 제공한다.

::표 9 | ILF 복잡도 매트릭스

(기능점수 변환: 낮음 = 7, 보통 = 10, 높음 = 15)

RET 개수 \ DET 개수	1~19개	20~50개	51개 이상
1개	낮음	낮음	보통
2~5개	낮음	보통	높음
6개 이상	보통	높음	높음

::표 10 | EIF 복잡도 매트릭스

(기능점수 변환: 낮음 = 5, 보통 = 7, 높음 = 10)

RET 개수 \ DET 개수	1~19개	20~50개	51개 이상
1개	낮음	낮음	보통
2~5개	낮음	보통	높음
6개 이상	보통	높음	높음

그림 4의 직원 정보 ILF와 그림 5의 주문 정보 ILF의 경우, 표 9의 기준에 의해 복잡도와 데이터 기능 점수를 측정한 결과는 표 11과 같다. 이 경우 외부 애플리케이션 데이터를 사용하지 않아 EIF 복잡도 매트릭스가 적용되지 않았다.

:: 표 11 | 직원 정보와 주문 정보의 데이터 기능 점수

ILF	RET	DET	복잡도	데이터 기능점수
직원 정보	3	14	낮음	7
주문 정보	3	15	낮음	7

트랜잭션 기능 점수 측정

트랜잭션 기능이란 애플리케이션이 사용자에게 제공하는 기능을 의미한다. 트랜잭션 기능은 EI(외부 입력), EQ(외부 조회), EO(외부 출력)로 구분된다. EI는 애플리케이션 안으로 입력되는 데이터(ILF를 유지하기 위해)나 제어 정보(시스템의 동작을 변경시키는)를 처리하는 것이다. EQ는 사용자의 요청(조회 또는 검색)에 따라 애플리케이션 외부로 ILF나 EIF의 정보를 내보내는 것이다. EO는 데이터나 제어 정보의 검색이 아닌 처리 로직을 가지고 데이터를 애플리케이션 밖으로 내보내는 것이다. 다시 말해, EQ가 있는 그대로의 정보를 보여주는 것이라면, EO는 저장된 정보를 가공 처리하여 보여주는 것이다.

예를 들어, 상품에 대한 상세정보를 조회하는 경우는 ILF를 유지(추가, 수정, 삭제)하지 않고 단지 외부로 보여주는 것이므로 EQ에 해당된다. 상품의 상세정보를 조회한 상태에서 상품의 수량을 입력하고 장바구니담기를 실행하는 경우, 새로운 상품을 장바구니목록에 추가함과 동시에 장바구니에 담긴 상품들의 수량과 가격을 수학적으로 계산하여 보여주므로 EO에 해당된다.

트랜잭션 기능을 식별하려면 먼저 단위 프로세스를 측정해야 한다. 단위 프로세스란 사용자에게 의미 있는 활동의 최소 단위이다. 단위 프로세스는 독립적이며, 애플리케이션의 비즈니스를 일관성 있는 상태로 유지한다. 측정된 단위 프로세스를 토대로 앞서 설명한 세 가지 유형의 트랜잭션 기능을 식별한다. 단위 프로세스로부터 식별될 수 있는 세 가지 트랜잭션 기능에 대해서 간략하게 정리하면 표 12와 같다.

:: 표 12 | 트랜잭션 기능의 주요 의도(primary intent)

트랜잭션 기능	주요 의도
EI	• ILF를 유지(추가, 수정, 삭제)하거나 시스템의 동작을 변경한다.
EO	• 사용자에게 데이터 및 제어정보를 처리 로직 혹은 검색을 통해 정보를 제공 • 처리 로직은 적어도 하나 이상의 수식 혹은 계산이 포함되어야 하며, 하나 이상의 ILF를 유지 혹은 시스템의 활동을 변경하는 데이터를 생성
EQ	• 사용자에게 데이터 및 제어정보를 ILF 혹은 EIF를 검색하여 정보를 제공 • 처리 로직은 수식 및 계산을 포함할 수 없으며, 데이터를 생성할 수 없음 • 처리 중에 ILF를 유지하거나, 시스템의 활동을 변경할 수 없음

트랜잭션 기능을 식별하기 위해서는 기능 요구사항이 어떠한 과정에 의해서 실현되는가를 분석하는 것이 좋다. 예를 들어, 유스케이스 시나리오는 각 기능 요구사항이 어떠한 과정을 거쳐 실행되는지를 설명하고 있다. 유스케이스 시나리오에서는 사용자가 시스템을 통해 어떠한 요청을 하고, 시스템은 어떤 결과를 보여주는지에 대한 정보가 포함되어 있기 때문에, 트랜잭션 기능을 분석하는 데 유용하게 사용될 수 있다.

위의 세 가지 트랜잭션 기능들을 계산하기 위해서 고려해야 하는 규칙들이 존재하며, 데이터 기능과 마찬가지로 트랜잭션 기능 또한 복잡도를 고려하여 최종 기능 점수를 산정해야 한다. 각 트랜잭션 기능들에 대한 계산 규칙과 복잡도 계산 방법들에 대해서 설명하고자 한다.

EI는 애플리케이션 경계 밖에서 내부로 데이터가 들어오는 경우를 말한다. EI를 식별하는 규칙은 다음과 같이 정리할 수 있다[한국전산원 2001].

- EI를 통해서 ILF에 있는 최소한 하나의 데이터가 유지(추가, 수정, 삭제)되어야 한다.
- EI를 처리하는 로직이 애플리케이션 내 다른 EI의 처리 로직과 구분될 수 있는 유일한 것이어야 한다.
- 해당 EI의 DET가 다른 EI의 DET와 구분되어야 한다.
- EI에서 참조한 ILF나 EIF가 다른 EI가 참조하고 있는 데이터 파일과 구분되어야 한다.
- 제어 정보는 애플리케이션 경계 밖에서 들어온다.
- 제어 정보는 애플리케이션의 요구사항을 준수하는지 보증하기 위해 사용자에게 의해 명시화되어야 한다.

위의 규칙에 따라 식별된 EI에 대해서 데이터 요소 타입(DET)과 참조 파일 타입(FTR: File Type Referenced)의 수를 기준으로 복잡도를 결정한다. DET에 대해서는 앞서 데이터 기능 점수 산정 방법을 소개할 때 설명한 바 있다. FTR은 간단하게 참조 파일이라고도 부르며, EI에 의해서 유지되거나 읽히는 ILF와 EI에 의해서 읽히는 EIF의 총 개수를 의미한다. EI의 복잡

도 측정 시 FTR의 계산 규칙은 다음과 같다[한국전산원 2001].

- EI의 기본 프로세스에 의해 유지되는 각 ILF에 대해서 하나의 FTR로 계산한다.
- EI의 처리 동안 참조되는 ILF나 EIF 각각에 대해서 하나의 FTR로 계산한다.
- EI에 의해 유지(추가, 변경, 삭제)되면서 동시에 참조되는 각 ILF는 하나의 FTR로만 계산한다.

이어서 EI의 복잡도 측정 시 DET의 계산 규칙은 다음과 같다.

- EI 프로세스 완료에 필요하면서 사용자가 식별 가능하고 반복되지 않는 필드를 하나의 DET로 계산한다.
- EI 수행 중 시스템에 의해 검색되거나 새로 생성되어 ILF에 저장되었지만, 애플리케이션 경계를 넘지 않는 것은 DET로 측정하지 않는다. 시스템에 의해서 생성된 날짜, 고유 번호 등이 그 예이다.
- 주소 라인처럼 물리적으로는 여러 필드로 저장되었으나, 단일 정보로 사용자가 요구하는 논리적 필드는 하나의 DET로 계산한다.
- 처리 중 발생한 에러 및 처리 완료를 표시하거나, 처리의 계속 여부 확인 등의 애플리케이션 경계 밖으로 내보내는 시스템 메시지는 하나의 DET로 계산한다.
- 동일한 처리 로직을 실행시키는 여러 가지의 방법은 동일한 액션으로 간주하여 하나의 DET로 계산한다.

표 13은 FTR과 DET의 개수에 의해 EI의 복잡도가 어떻게 결정되는지를 보여준다. EI의 복잡도에 따라 해당 트랜잭션의 기능 점수가 정해진다.

표 13 | EI 복잡도 매트릭스

(기능 점수 변환: 낮음 = 3, 보통 = 4, 높음 = 6)

RET 개수 \ DET 개수	1~4개	5~15개	16개 이상
0~1개	낮음	낮음	보통
2개	낮음	보통	높음
3개 이상	보통	높음	높음

EO는 애플리케이션의 경계 밖으로 나가는 데이터나 제어 정보를 생성하는 애플리케이션의 단위 프로세스로서 데이터나 제어 정보의 단순 검색이 아닌 처리 로직을 통해 사용자에게 정보를 제시한다. 처리 로직은 최소한 하나의 수학적식이나 계산을 포함해야 하고, 파생된 데

이터를 생성하고, 하나 이상의 ILF를 유지한다.

EQ는 애플리케이션의 경계 밖으로 데이터나 제어 정보의 검색 결과를 내보내는 애플리케이션의 단위 프로세스로서 ILF나 EIF로부터 사용자에게 정보를 제공하는 것이다. 처리 로직은 수학적식이나 계산을 포함하지 않고, 유도된 데이터를 생성하지 않아야 한다. 또한 처리 중에 ILF가 유지되지 않아야 한다.

앞서 세 가지의 트랜잭션 기능에 대해 설명할 때, 제어 정보나 파생 데이터와 같은 용어를 사용하였다. 참고로 제어 정보란 애플리케이션의 기본 프로세스에 영향을 주기 위해 애플리케이션에 의해 이용되는 데이터이다. 제어 정보는 애플리케이션에 의해 사용자나 다른 애플리케이션에 송신된다. 예를 들어, 사용자가 명시한 기능 요구사항을 준수하는지 보증하기 위해 송신되는 데이터로, 특정 내부 조건이 설정되었음을 알리는 메시지와 같다. 파생 데이터는 추가적인 데이터를 생성하기 위해 기본 데이터의 변환을 통해 생성된다.

EO와 EQ 역시 DET와 FTR의 수를 기준으로 복잡도를 측정하여 최종적인 기능 점수를 산정한다. 먼저 EO의 FTR 계산 규칙을 정리하면 다음과 같다.

- EO의 처리 동안 참조되는 ILF나 EIF 각각에 대해서 하나의 FTR로 계산한다.
- EO에 의해서 유지되는 각 ILF에 대해서 하나의 FTR로 계산한다.
- EO에 의해 유지되면서 동시에 참조되는 각 ILF에 대해서 오직 하나의 FTR로 계산한다.

이어서 EQ의 FTR 계산 규칙은 다음과 같다.

- EQ의 처리 동안 참조되는 ILF나 EIF 각각에 대해서 하나의 FTR로 계산한다.

EO와 EQ에 대해서 FTR을 모두 계산했으면, 다음은 DET에 대해서 계산한다. EO와 EQ에 대해서 공통적으로 적용할 수 있는 DET 계산 규칙은 다음과 같다[한국전산원 2001].

- 애플리케이션 경계 밖에서 입력하고, 사용자가 식별 가능하고, 반복되지 않으며, 언제, 어떤 데이터를 어떻게 검색 및 생성할지를 결정하는 필드를 DET로 계산한다.
- 애플리케이션 경계를 넘어 출력되고, 사용자가 식별 가능한 반복되지 않는 필드를 하나의 DET로 계산한다.
- 애플리케이션 경계를 기준으로 입력 및 출력이 이루어지는 DET는 해당 단위 프로세스에서 한 번만 계산한다.
- 처리 중 발생한 에러 및 처리 완료를 표시하거나, 처리의 계속 여부 확인 등의 애플리케이션 경계 밖으로 내보내는 시스템 메시지는 하나의 DET로 계산한다. 예를 들어, 새로운 직원 정보 입력 시, 직원 정보를 화면에 모두 기입한 후 저장 버튼을 클릭하면 시스

템은 정상적인 경우 ‘직원 정보가 저장되었습니다’와 같은 메시지를, 정상적이지 않은 경우 ‘직원 정보 저장에 실패하였습니다’와 같은 메시지를 출력할 것이다. 이 경우 해당 트랜잭션의 결과를 보여주는 메시지도 하나의 DET로 계산한다. 단, 정상적인 경우와 정상적이지 않은 경우에 대해서 별개의 DET로 계산하지는 않는다.

- 리포트 제목, 스크린 ID, 열 표제(column heading), 필드 제목을 포함하는 문자 상수는 계산하지 않는다. 예를 들어, 직원 정보의 각 입력 필드를 설명하기 위한 필드 이름에 대해서는 DET로 계산하지 않는다.
- 물리적으로는 여러 필드로 저장되었지만 사용자에게 의해 단일 정보로 요구되는 논리적 필드에 대해서는 하나의 DET로 계산한다.
- 그래픽 디스플레이에서 각 유형의 레이블과 수치에 대해 각각 하나의 DET로 계산한다. 예를 들어, 파이 차트는 범주와 백분율로 표현되므로 2개의 DET로 계산된다.

EO와 EQ 각각에 대해서 FTR과 DET를 모두 계산하면 다음과 같은 복잡도 매트릭스를 통해서 각 트랜잭션의 기능 점수를 산정하게 된다.

::표 14 | EO의 복잡도 매트릭스

(기능 점수 변환: 낮음 = 4, 보통 = 5, 높음 = 7)

RET 개수 \ DET 개수	1~5개	6~19개	20개 이상
0~1개	낮음	낮음	보통
2~3개	낮음	보통	높음
4개 이상	보통	높음	높음

::표 15 | EQ의 복잡도 매트릭스

(기능 점수 변환: 낮음 = 4, 보통 = 5, 높음 = 7)

RET 개수 \ DET 개수	1~5개	6~19개	20개 이상
0~1개	낮음	낮음	보통
2~3개	낮음	보통	높음
4개 이상	보통	높음	높음

예를 들어, 앞의 그림 4에서 표현한 ER 다이어그램의 직원 정보를 애플리케이션에 저장하고자 한다면, 트랜잭션 기능의 유형은 EI에 해당된다. FTR의 개수는 트랜잭션에 의해 참조되거나 유지되는 ILF의 개수이므로 1개로 계산된다. DET의 개수는 그림 4의 엔티티들에 포함된 기본 속성들과 트랜잭션 실행에 필요한 기능 버튼(예를 들어, 저장 버튼), 트랜잭션 실행 후 결과를 보여주는 메시지(예를 들어, ‘직원 정보가 성공적으로 저장되었습니다’) 등에 의해 결정된

다. 직원 정보 저장의 경우, FTR의 개수가 1개이고 DET의 개수가 16개이므로 표 13에 의해 EI의 복잡도는 보통으로 처리되어 기능 점수는 4점이 되었다. DET 측정에 사용된 속성들은 밑줄을 그어 나타내었다.

::표 16 | EI의 복잡도

기능명	기능 유형	측정된 DET	FTR	DET	복잡도	기능점수
직원 정보 저장	EI	Employee Number(PK) Name Address City State Zip Salary Hourly Wage	1	16	보통	4
		Name Address City State Zip Relation 저장버튼 처리결과메시지				

지금까지 소개한 데이터 및 트랜잭션 기능 점수 산정 방법은 IFPUG에서 정의한 정규 기능 점수 방법(줄여서 정규법이라 한다)에 해당된다. 이 방법은 개발 프로젝트가 진행 중에 있거나 개발이 완료된 시점에서 적용하여 보다 정확한 규모를 측정할 수 있게 하지만, 예산 수립이나 사업 제안 단계에서는 적용하기에 무리가 따른다. 그 이유는 예산 수립이나 사업 제안 단계에서는 기능 수준만 도출해 내고 분석, 설계 단계를 거치면서 비로소 기능의 속성까지 상세하게 도출될 수 있기 때문이다. 다시 말해, 개발 프로젝트 초기에 RET, DET, FTR의 개수를 고려하여 기능의 복잡도를 결정하는 것은 현실적으로 어려운 점이 있다.

따라서 소프트웨어 사업 대가 기준에서는 사업 초기 각 기능의 복잡도와 가중치를 결정하기 어려운 현실을 반영하여 표 17과 같은 평균 복잡도 가중치를 사용하도록 하였다. 데이터 기능과 트랜잭션 기능의 수가 도출된 후, 복잡도를 계산하는 과정을 생략하고, 제시된 평균 복잡도 가중치를 바로 기능 수에 곱하여 최종 기능 점수를 구하는 방법이다. 평균 복잡도 가중치는 과거에 수행된 소프트웨어의 기능 점수 산정 결과를 통계적으로 분석하여 ILF, EIF, EI, EO, EQ에 대한 복잡도의 평균값을 계산한 것이다.

트랜잭션 기능의 복잡도를 보면 EO가 가장 높고 EI와 EQ의 경우 상대적으로 낮은 것을 볼

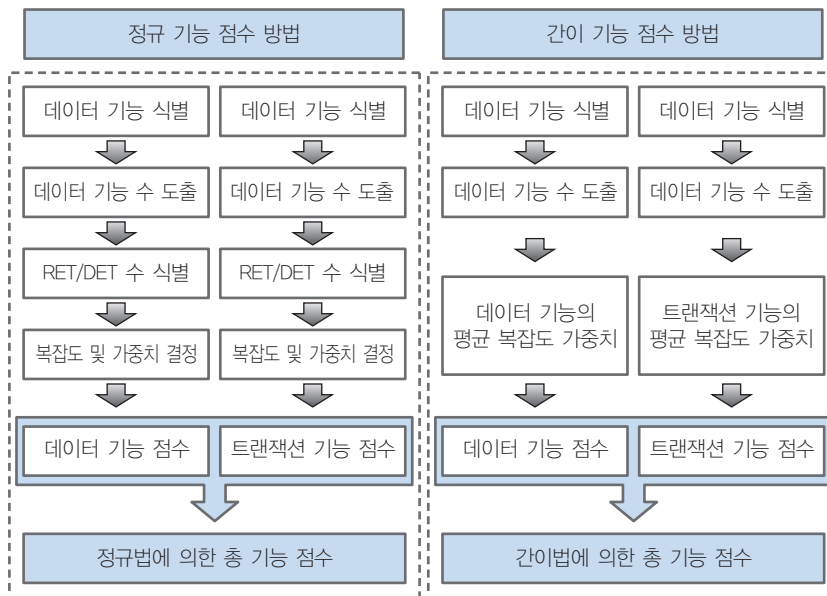
수 있다. EO의 경우, 내부 알고리즘의 복잡도가 EI나 EQ에 비해 상대적으로 높기 때문에 복잡도 수치가 높다. 반면, EI와 EQ는 일반적으로 데이터베이스 테이블의 내용을 단순히 추가하거나 삭제, 조회하는 등의 기능만 구현하면 되기 때문에 그 복잡도가 상대적으로 낮다.

데이터 기능의 경우 ILF가 EIF보다 높은 것을 볼 수 있는데, 이는 ILF가 참조될 뿐만 아니라 유지되어야 하는 대상이기 때문이다. EIF는 참조의 대상이 외부 애플리케이션 경계 내에 존재하기 때문에 참조 이상의 기능을 수행할 수 없는 대상이므로 복잡도를 낮게 잡았다고 볼 수 있다.

:: 표 17 | 지식경제부 소프트웨어 대가 기준의 평균 복잡도 가중치

기능 유형	평균 복잡도 가중치
EI	4.0
EO	5.2
EQ	3.9
ILF	7.5
EIF	5.4

평균 복잡도 가중치를 적용하여 기능 점수를 산정하는 방식을 간이 기능 점수 방법, 줄여서 간이법이라 한다. 정규법과 간이법을 이용한 기능 점수 산정 절차를 정리하면 그림 6과 같다.



:: 그림 6 | 정규 기능 점수 방법과 간이 기능 점수 방법의 절차 비교

출처: 송기호, “기능점수 산정 및 활용 방안”

③ 프로세스 기반 공수 추정 모델

프로젝트의 원가를 추정하는 가장 정확한 방법은 상향식 추정(bottom-up estimating)이다. 프로세스 기반 공수 추정은 소프트웨어 프로젝트에서 채택한 프로세스의 각 단계(예를 들어, 분석, 설계, 구현)별 활동에 소요되는 기간과 투입 인력을 예측하여 비용을 계산한다. 즉, 개발 라이프 사이클을 작은 단위의 활동들로 분해하고 각 작업을 수행하는 데 드는 노력을 측정한다. 이 방법은 과거 프로젝트 기록이나 경험을 바탕으로 각 단계의 입력물, 출력물, 표준 작업 시간, 필요한 자원을 추정하면 된다. 이러한 상향식 추정은 비용과 시간이 많이 소요되지만 가장 합리적이고 정확한 결과를 얻을 수 있다.

시간 산정과 마찬가지로 프로젝트 수행에 요구되는 작업을 산출물 단위로 나눈 WBS는 프로젝트의 비용을 추정할 때 긴요하게 사용될 수 있다. 프로세스 기반 공수 추정 모델은 WBS의 작업 패키지별로 원가를 계산하고 상위 단계로 올라가며 원가를 합하여 전체 비용을 추정하는 방법이다. 원가 산정(cost estimation)은 WBS에 기술된 각각의 작업 패키지에 요구되는 자원의 원가 산정치를 구한 후 상위 수준에서 요약 집계하는 것이다.

소프트웨어 프로젝트의 경우 프로젝트 초기 단계에서 프로젝트를 성공적으로 완수하기 위해 요구되는 모든 작업(WBS)을 규명할 수는 있다. 그러나 프로젝트 초기 단계에서는 요구사항과 프로젝트의 범위가 불확실하여 정확히 정의하기가 어렵고 노력이나 비용 추정과는 상관없이 WBS를 작성하는 경우가 많아 소프트웨어 프로젝트에 상향식 추정 기법을 적용하기 어렵다. 하지만 프로젝트가 점진적으로 진행됨에 따라 WBS에 요구되는 노력이나 비용 추정이 가능하게 되는 상황에서 상향식 기법을 추가하여 적용하면 보다 정확하게 원가를 추정하고 측정할 수 있다.

④ 기능 점수 산정 방식

기능 점수 산정 방식은 크게 정규법과 간이법으로 구분한다. 이 절에서는 앞의 두 가지 방식을 이용하여 인터넷 쇼핑몰 시스템의 기능 점수를 측정하는 과정을 설명한다.

간이법 적용

간이법이 정규법과 다른 점은 앞서도 설명했듯이 데이터 기능과 트랜잭션 기능의 복합도 분석 시 사용되는 RET, DET, FTR의 측정을 배제하는 것이다. 이 경우, 데이터 및 트랜잭

선 기능의 복잡도를 개발자들의 판단에 의해 결정할 수 있지만, 복잡도 파악이 어려울 경우에는 표 11의 평균 복잡도를 적용한다.

간이법을 사용할 경우, 데이터 기능과 트랜잭션 기능의 수를 계산하고 계수된 모든 기능의 수에 위의 평균 복잡도를 곱하여 미조정 기능 점수를 구한 후, 보정 계수를 적용하여 최종 조정 기능 점수를 구한다. 지금부터 간이법을 적용하여 간단한 애플리케이션에 대한 기능 점수를 산정하는 과정에 대해서 알아보자.

인터넷 쇼핑몰 시스템에서 식별한 몇몇 기능 요구사항(유스케이스)들을 대상으로 데이터 기능과 트랜잭션 기능의 수를 먼저 파악하고, 그 결과를 토대로 각 기능에 대한 평균 복잡도와 애플리케이션의 전반적인 보정 계수를 적용하여 전체 기능 점수를 산정하는 과정에 대해서 설명하고자 한다. 데이터 기능 및 트랜잭션 기능에 대한 복잡도는 지식경제부에서 고시한 표 17의 평균 복잡도를 적용한다.

인터넷 쇼핑몰 애플리케이션의 기능 요구사항을 바탕으로 데이터 기능과 트랜잭션 기능의 수를 구한 결과가 표 18에 나타나 있다. 각 기능의 유형을 결정하고 기능 유형별로 표 17의 평균 복잡도를 적용하여 기능 점수를 구한다. 이렇게 모든 기능들에 대해서 기능 점수를 구하고 그 결과들을 합산하여 총 미조정 기능 점수를 산정한다. 여기에 여러 가지 측면을 고려한 보정 계수를 적용하면 최종적인 애플리케이션 기능 점수를 산정할 수 있다.

::표 18 | 기능 점수표(간이법 적용)

업무명	기능명	기능 유형	평균 복잡도에 의한 기능 점수
회원 관리	회원가입	EI	4.0
	사용자	ILF	7.5
카테고리 관리	카테고리등록	EQ	3.9
		EI	4.0
	카테고리	ILF	7.5
상품 관리	상품등록	EQ	3.9
		EI	4.0
	상품목록조회	EQ	3.9
	상품상세조회	EQ	3.9
	상품	ILF	7.5
	장바구니담기	EO	5.2
주문결제 관리	상품주문	EI	4.0
	주문결제	EI	4.0
		EQ	3.9

(계속)

업무명	기능명	기능 유형	평균 복잡도에 의한 기능 점수
주문결제 관리	온라인입금처리	EQ	3.9
		EI	4.0
		EO	5.2
	주문결제	ILF	7.5
	승인결과	EIF	5.4

표 18의 기능 점수 표에서 기능들의 점수를 합산하면 93.2점이 된다(인터넷 쇼핑몰의 일부 기능만을 대상으로 한 점수이므로 전체 기능 점수는 더 높을 수 있다). 이 결과에 보정 계수를 적용하면 애플리케이션의 최종 기능 점수가 된다. 앞서 설명했듯이 최종 기능 점수(조정 기능 점수)를 얻기 위해 적용해야 하는 보정 계수는 규모 보정 계수, 애플리케이션 유형 보정 계수, 언어 보정 계수, 품질 및 특성 보정 계수이다.

먼저 규모 보정 계수는 기능 점수가 300점 이상인 경우 보정 계수 산출식을 이용하지만, 300점 미만인 경우 0.65로 정한다. 사례의 애플리케이션은 총 기능 점수가 93.2점이므로 규모 보정 계수는 0.65가 된다.

애플리케이션 유형 보정 계수는 표 6의 내용을 참조한다. 애플리케이션의 특성을 고려하여 애플리케이션 유형의 비중을 할당한 다음, 비중에 따른 보정 계수의 합을 구하도록 한다. 이 사례의 애플리케이션은 업무 처리용 애플리케이션에 해당된다고 볼 수 있으며, 비중은 100%를 할당한다. 따라서 애플리케이션 유형 보정 계수는 1.0이 된다.

언어 보정 계수는 애플리케이션 개발에 필요한 언어가 무엇인지를 고려하여 결정한다. 앞서 설명했듯이 언어 보정 계수는 발주자(또는 고객)가 특정 언어를 요구하는 경우, 사용된 언어의 비율에 따라 소프트웨어 개발 단계 중 구현과 시험 단계에만 적용한다. 만약 인터넷 쇼핑몰에 대해서 고객이 자바 기반의 웹 애플리케이션으로 개발해 주기를 원한다고 가정하면, 개발 업체는 Java 및 JSP와 같은 언어를 활용하게 될 것이다. 두 언어가 사용되는 경우, 각각의 사용 비중을 따져 봐야 하는데, 이는 비슷한 유형의 애플리케이션을 개발한 경험을 바탕으로 결정될 것이다. 여기서는 Java와 JSP의 사용 비중이 각각 80%와 20% 정도임을 가정한다. 표 7의 내용을 참조하면, 이 사례의 애플리케이션에 대한 언어 보정 계수는 $1.2 \times 0.8 + 0.8 \times 0.2$ 의 산출식에 의해 1.12가 될 것이다.

마지막으로 품질 및 특성 보정 계수를 고려하도록 한다. 품질 및 특성 보정 계수는 표 8의 내용을 참조한다. 각 보정 요소별 판단 기준에 따라서 영향도를 평가하고, 모든 보정 요소에 대해서 총 영향도를 계산한다. 총 영향도가 구해지면 정해진 산출식에 따라 품질 및 특성 보정 계수를 계산한다. 표 8의 기준에 따라서 인터넷 쇼핑몰의 품질과 특성을 평가해 보면,

분산 요소는 1의 영향도, 성능 요소는 1, 신뢰성 요소는 1, 다중 사이트 요소는 0이 될 것이다. 따라서 총 영향도는 3이 되며, 품질 및 특성 보정 계수는 산출식 $0.025 \times 3 + 1.0$ 에 의해 1.075가 된다.

지금까지 사례에서 소개한 인터넷 쇼핑몰의 미조정 기능 점수와 보정 계수의 측정에 대한 과정을 소개하였다. 애플리케이션의 최종 조정 기능 점수는 미조정 기능 점수와 보정 계수들의 곱으로 얻을 수 있다. 단, 언어 보정 계수의 경우는 구현과 시험 단계에만 적용해야 한다는 기준이 있으므로, 개발 원가 산정 시 반드시 고려해야 한다.

조정 기능 점수 산정에 필요한 모든 요소를 확보했으면, 쇼핑몰 애플리케이션의 개발 프로젝트를 진행하기 위해 총 얼마의 비용이 드는지 확인할 필요가 있다. 지식경제부에서 고시한 대가 산정 기준을 보면 기능 점수 1점에 대하여 각 개발 단계별 개발 단가가 명시되어 있다. 그 내용은 표 19와 같다.

표 19 | 개발 단계별 단가표

단계	단계별 단가
분석	94,511원
설계	119,382원
구현	159,177원
시험	124,357원

표 19의 단가표를 기준으로 프로젝트의 총 개발 원가를 산출한 결과가 표 20에 나타나 있다. 각 단계별 단가와 미조정 기능 점수, 여러 보정 계수의 곱을 통해 단계별 개발 원가를 구할 수 있으며, 이를 통해 개발 프로젝트에 대한 총 개발 원가를 산출할 수 있다. 표 20에서 언어 보정 계수는 원칙에 따라 구현과 시험 단계에서만 적용되었음을 다시 한 번 알린다.

::표 20 | 개발 원가 산출 결과

단계	단계별 단가	미조정 기능 점수	보정 계수				개발 원가
			규모	애플리케이션 유형	언어	품질 및 특성	
분석	94,511원	93.2	0.65	1.0	1.0	1,075	6,154,887원
설계	119,382원				1.0		7,774,573원
구현	159,177원				1.2		12,439,396원
시험	124,357원				1.2		9,718,275원
계	497,427원						36,087,131원

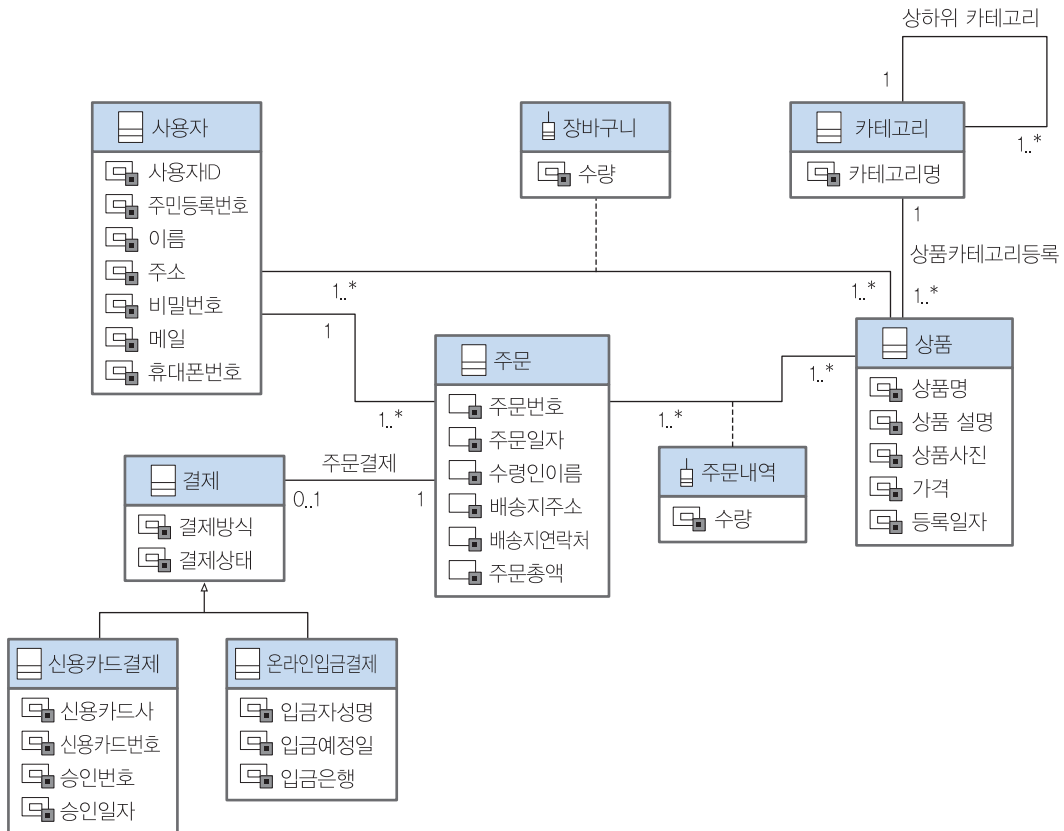
정규법 적용

정규법을 적용하여 기능 점수를 측정하는 과정을 설명하도록 한다. 정규법에서는 RET, DET, FTR의 개수를 통해 데이터 기능과 트랜잭션 기능의 복잡도를 측정하는 과정이 필요하다. 먼저 데이터 기능 점수를 측정하기 위해서 RET와 DET의 개수를 계산해야 한다. RET와 DET의 개수를 계산하기 위해서는 분석 단계의 데이터 모델을 참조하는 것이 일반적이다. 이번 장에서 예로 든 인터넷 쇼핑몰의 분석 단계 데이터 모델은 그림 7과 같다. 그림 7은 클래스 다이어그램으로 표현한 데이터 모델이며, 경우에 따라 ER 다이어그램을 사용하기도 한다.

RET의 개수를 구하기 전에 클래스 다이어그램에서 ILF가 될 수 있는 클래스(엔티티 타입)가 무엇인지를 식별하는 것이 필요하다. ILF로 식별되기 위한 조건은 앞서 설명한 바 있으며, 기본적으로 사용자가 식별할 수 있는 독립적인 클래스여야 한다. 클래스가 독립적으로 존재할 수 있는 것인지 판별하기 위해 클래스들 사이의 관계를 고려해야 한다. 두 클래스가 관계를 맺을 때, 어느 한 클래스가 다른 클래스에 종속되지 않고 독립적으로 존재할 수 있다면, 두 클래스를 별도의 ILF로 식별한다. 반대로 어느 한 클래스가 다른 클래스에 종속된다면, 두 클래스를 묶어 하나의 ILF로 식별한다. 이는 두 클래스 사이의 참여 제약 조건과 상관없이 일반적으로 적용할 수 있는 원칙이지만, 특별한 경우에는 종속된 클래스도 별도의 ILF로 간주될 수 있다.

두 클래스가 다대다 관계를 맺게 되는 경우, 두 클래스 사이의 관계 정보를 나타내기 위한 연관 클래스(연관 엔티티 타입)가 필요하다. 연관 클래스는 일반적으로 별도의 ILF로 식별되지 않는다. 왜냐하면 연관 클래스는 독립적으로 존재할 수 없고, 단지 양쪽 두 클래스의 관계 정보만을 포함하기 때문이다. 이 경우, 양쪽의 클래스들은 연관 클래스를 포함하여 하나의 ILF로 식별하는 것이 기본 원칙이다. 만약 연관 클래스가 어느 한쪽의 클래스에만 포함되어야 한다는 요구사항이 있다면, 한쪽 클래스에만 포함시켜 ILF로 식별하도록 한다. 양쪽 두 클래스의 존재 여부와 상관없이 어떤 목적을 위해 그 정보를 유지해야 한다는 요구사항이 있다면

연관 클래스도 별도의 ILF로 식별하는 것이 가능하다.



::그림 7 | 인터넷 쇼핑몰의 데이터 모델(클래스 다이어그램)

이러한 원칙들을 기반으로 그림 7의 클래스 다이어그램에서 ILF로 식별될 수 있는 클래스들은 사용자, 카테고리, 상품, 주문이다.

주문 클래스는 사용자와 일대다 관계를 맺으면서 사용자에게 종속되는 클래스로 고려될 수 있으나, 사용자 클래스가 삭제되더라도 주문 정보가 유지되어야 한다면 별도의 ILF로 식별해야 할 것이다. 인터넷 쇼핑몰의 운영자 관점에서 보면 사용자 정보가 삭제된다 하더라도 총 판매 기록을 유지 보관해야 하므로 주문 정보 역시 하나의 ILF로 식별하는 것이 바람직할 것이다.

다대다 관계인 사용자와 상품을 연결하는 장바구니는 연관 클래스이다. 연관 클래스는 일반적으로 ILF로 식별되지 않으며, 양쪽의 연결된 클래스에 포함된다. 따라서 기본 원칙에 의하면, 사용자와 장바구니, 상품과 장바구니를 묶어 각각 하나의 ILF로 식별해야 한다. 하지만 장바구니는 어떤 상품에 포함될 수 있는 개념이 아니라 어떤 사용자가 소유한 것인지가 중요하기 때문에 사용자와 장바구니를 묶어 하나의 ILF로 식별하고 상품은 따로 떼어내어 ILF로

식별하는 것이 합당하다.

카테고리 클래스는 그 자체로 재귀 연관 관계를 맺고 있다. 카테고리는 여러 하위카테고리를 포함할 수 있기 때문이다. 하위카테고리는 바로 상위카테고리와 관계를 맺게 된다. 카테고리 클래스를 풀어서 표현하면 카테고리와 카테고리가 관계를 맺고 있는 형태가 될 것이다. 일대다 관계로서 하위카테고리는 상위카테고리에 종속되는 개념이므로 두 카테고리를 묶어 하나의 ILF로 식별하도록 한다.

카테고리와 상품은 일대다 관계를 맺고 있으며 상품은 카테고리에 포함되는 개념이다. 따라서 상품 클래스가 카테고리 클래스에 종속된다고 볼 수도 있으나, 카테고리가 삭제된다고 해서 그것에 포함된 모든 상품 클래스들이 삭제될 필요는 없으므로 각각을 하나의 ILF로 식별하는 것이 좋다. 만약 카테고리를 삭제하려고 하는 경우, 그것에 포함된 상품들은 다른 카테고리에 이동시키고 삭제하는 것이 바람직하다.

주문과 상품은 다대다 관계를 맺고 있다. 따라서 주문과 상품 클래스 사이에 주문내역이라는 연관 클래스가 존재한다. 원칙에 따르면 주문과 주문내역, 상품과 주문내역을 묶어 각각 별도의 ILF로 식별하는 것이 맞지만, 주문내역 역시 주문 클래스에 포함되어야 할 정보이지 상품 클래스에 포함될 정보는 아니다. 따라서 주문과 주문내역을 묶고 상품은 역시 독립적인 ILF로 식별하도록 한다.

마지막으로 주문과 결제 클래스의 관계를 보면 일대일 관계라는 것을 알 수 있다. 단, 결제는 주문에 대해 종속적이며 선택적으로 존재할 수 있다. 이 경우, 주문과 결제 클래스를 하나로 묶어 ILF로 식별하도록 한다. 신용카드결제와 온라인입금결제는 결제 클래스와 상속 관계를 맺고 있으므로 자동으로 포함된다. 결과적으로 주문 클래스는 결제와 주문내역까지 포함하는 ILF로 식별된다.

지금까지는 데이터 모델을 참조하여 ILF를 식별하는 과정을 설명하였다. 데이터 모델을 참조하는 방법 외에도 사용자의 비즈니스 룰, 즉 기능 요구사항의 수행과정을 통해 식별되는 경우도 있다. 인터넷 쇼핑몰의 경우, 주문 결제 요구사항을 실행하기 위해 신용카드 인증 회사로부터 승인 결과를 조회하는 과정이 존재한다. 이 경우, 승인 결과 정보는 데이터 모델에서는 표현되지 않았지만 외부 애플리케이션으로부터 조회하는 논리 파일에 해당되기 때문에 EIF로 식별한다.

모든 ILF와 EIF를 식별했으면, 각 ILF를 구성하는 RET와 DET의 개수를 계산해야 한다. RET는 일반적으로 ILF로 식별된 클래스에 대응된다. 다시 말하면, 일반적으로 하나의 클래스는 한 개의 RET를 갖는다는 말이다. 여기에 다른 클래스와의 관계, 연관 클래스 포함 여부 등을 고려하여 RET의 개수를 계산한다. 앞의 ILF 식별과정에서 클래스들의 관계를 논하면서 각 ILF에 포함된 클래스들이 무엇인지 설명한 바 있다. 각 ILF에 포함된 클래스들의 수를 알

면 쉽게 RET의 개수를 구할 수 있다.

사용자는 장바구니를 포함하여 ILF로 식별되었으므로 이 경우 RET의 개수는 2가 된다. 카테고리 클래스는 하위 카테고리와의 연관 관계를 맺고 있으므로 이 경우 카테고리와 하위 카테고리가 각각 RET가 되어 RET의 개수는 2가 된다. 상품 클래스는 그 자체로 ILF로 식별되었기 때문에 RET의 개수는 1이 된다. 카테고리 클래스는 실질적으로 보면 카테고리와 하위 카테고리의 연관 관계로 표현되었지만 두 클래스의 타입이 같으므로 RET는 1로 계산한다. 주문 클래스의 경우는 주문내역, 결제(신용카드결제, 온라인입금결제) 클래스를 포함하여 ILF로 식별되었기 때문에 총 4개의 RET(주문, 주문내역, 신용카드결제, 온라인입금결제)로 계산된다.

각 ILF에 대해서 RET를 계산했으면, 마지막으로 각 ILF의 DET의 개수를 계산한다. DET는 ILF와 EIF를 구성하는 속성들이라고 보면 된다. DET 계산 시 적용된 몇 가지 원칙은 다음과 같다.

- 클래스들 사이의 관계를 나타내기 위해 사용되는 외래 키는 DET로 계산한다. 예를 들어, 카테고리와 상품은 일대다 관계를 맺으며, 이 경우 카테고리의 주 키인 카테고리명은 상품 클래스에 외래 키로 저장된다. 따라서 상품 ILF에 대해서 DET를 계산할 때 카테고리명을 포함시킨다. 다대다 관계인 경우, 양쪽 모두의 주 키를 외래 키로 포함시켜 계산하도록 한다.
- 클래스를 구성하는 속성 중 사용자가 입력하지 않고 시스템이 자동으로 생성하여 저장하는 속성(예를 들어, 생성일자, 주문번호 등)들은 DET로 계산하지 않는다.
- 여러 개의 필드로 구성되지만, 하나의 정보를 나타내는 경우(예를 들어, 주소, 전화번호, 이름 등)는 1개의 DET로 계산한다. 단, 검색이나 정렬 등의 이유로 각 필드를 따로 저장하려는 경우는 각각에 대해서 1개의 DET로 계산한다.

위의 원칙들에 의해서 DET의 개수를 모두 식별했으면, RET의 개수와 함께 복잡도를 측정하여 미조정 데이터 기능 점수를 산정한다. 표 9와 표 10의 복잡도 매트릭스를 통해 인터넷 쇼핑몰의 데이터 기능 점수를 산정한 결과는 표 21과 같다.

● 표 21 | 인터넷 쇼핑몰의 각 논리 파일에 대한 복잡도 측정 및 데이터 기능 점수

LF	RET	DET	복잡도	데이터 기능 점수
사용자	2	9	낮음	7
카테고리	2	2	낮음	7
상품	1	8	낮음	7
주문	4	18	낮음	7
신용카드승인결과(EIF)	1	4	낮음	5

트랜잭션 기능 점수를 계산하기 위해서는 FTR과 DET의 개수를 계산하여 복잡도를 측정해야 하며, 복잡도 결과에 따라 각 트랜잭션 기능의 점수가 부여된다. FTR은 해당 트랜잭션이 수행되는 과정에서 참조되거나 유지되는 ILF와 참조되는 EIF를 의미한다. DET는 데이터 기능 복잡도 측정에서 설명한 바 있다. 데이터 기능 측정 시에는 ILF를 구성하는 속성(또는 필드)들만을 다루었으나, 트랜잭션 기능을 측정할 때에는 트랜잭션 처리과정에서 보여지는 화면의 구성 요소들도 고려해야 한다. 대표적인 것이 버튼과 메시지이다. 버튼은 사용자가 시스템에 어떤 트랜잭션을 요청하는 경우 주로 사용되며, 메시지는 특정 트랜잭션의 처리 결과를 사용자에게 보여주는 것이다.

정규법을 이용하여 트랜잭션 기능들(간이법에서 측정되었던 것과 동일)의 복잡도와 점수를 계산한 결과가 표 22에 나타나 있다. 표 18과 다른 점은 트랜잭션 기능의 점수를 측정할 때, FTR과 DET의 개수를 이용했다는 것이다.

::표 22 | 인터넷 쇼핑물의 트랜잭션 기능에 대한 복잡도 측정 및 트랜잭션 기능 점수

업무명	기능명	기능 유형	FTR	DET	복잡도	트랜잭션 기능 점수
회원 관리	회원가입	EI	1	9	낮음	3
카테고리 관리	카테고리등록	EQ	1	2	낮음	4
		EI	1	4	낮음	3
상품 관리	상품등록	EQ	1	2	낮음	4
		EI	1	7	낮음	3
	상품목록조회	EQ	1	3	낮음	4
	상품상세조회	EQ	1	5	낮음	4
주문결제 관리	장바구니상품담기	EO	1	5	낮음	4
	상품주문	EI	1	5	낮음	3
	주문결제	EI	1	10	낮음	3
		EQ	1	1	낮음	4
	온라인입금처리	EQ	1	7	낮음	4
		EI	1	2	낮음	3
		EO	3	7	보통	5

참고로 각 트랜잭션 기능에서 DET의 개수가 계산된 근거는 다음과 같다.

:: 표 23 | 각 트랜잭션 기능의 DET 개수

기능명	기능 유형	DET
회원가입	EI	사용자ID, 주민등록번호, 이름, 주소, 비밀번호, 메일, 휴대폰번호, 회원가입버튼, 회원가입결과메시지
카테고리등록	EQ	카테고리조회버튼, 카테고리명
	EI	카테고리명, 상위카테고리명, 카테고리등록버튼, 카테고리등록결과메시지
상품등록	EQ	카테고리조회버튼, 카테고리명
	EI	카테고리명, 상품명, 상품설명, 상품사진, 가격, 상품등록버튼, 상품등록결과메시지 (등록일자는 시스템이 자동으로 생성/저장하는 필드이므로 DET로 계산하지 않음)
상품목록조회	EQ	카테고리명, 상품명, 가격
상품상세조회	EQ	상품명, 상품사진, 가격, 상품설명, 등록일자
장바구니상품담기	EO	장바구니담기버튼, 상품명, 수량, 가격, 총액
상품주문	EI	수령인이름, 배송지주소, 배송지연락처, 상품주문버튼, 상품주문결과메시지
주문결제	EI	결제방식라디오버튼, 신용카드사, 신용카드번호, 유효기간, 비밀번호, 할부기간, 입 금자성명, 입금예정일, 입금은행, 결제버튼
	EQ	결제승인결과메시지
온라인입금처리	EQ	입금자조회버튼, 주문번호, 주문일자, 주문총액, 입금자성명, 입금은행, 입금예정일
	EI	결제완료처리버튼, 결제처리완료메시지
	EO	이름, 수령인이름, 배송지주소, 배송지연락처, 상품명, 가격, 수량

지금까지 산정한 데이터 기능 점수와 트랜잭션 기능 점수의 합을 구하면 미조정 기능 점수가 된다. 표 21의 데이터 기능 점수와 표 22의 트랜잭션 기능 점수를 모두 합하면 정규법에 의한 미조정 기능 점수가 되며 그 결과는 84점이 된다. 앞서 간이법을 적용한 기능 점수의 결과와는 약간의 차이가 남을 알 수 있다. 미조정 기능 점수에 보정 계수를 곱하여 최종적인 기능 점수를 구하는 과정은 간이법과 동일하다. 정규법에 의해 구해진 기능 점수에 따라 프로젝트 개발 원가를 산출하면 다음과 같다.

:: 표 24 | 정규법에 의한 개발 원가 산출 결과

단계	단계별 단가	미조정 기능 점수	보정 계수				개발 원가
			규모	애플리케이션 유형	언어	품질 및 특성	
분석	94,511원	84	0.65	1.0	1.0	1.075	5,547,323원
설계	119,382원				1.0		7,007,126원
구현	159,177원				1.2		11,211,472원
시험	124,357원				1.2		8,758,961원
계	497,427원						32,524,882원

간이법에 의한 기능 점수의 규모가 정규법에 의한 것보다 다소 크게 나온 이유는 평균 복잡도의 크기와 실제 측정된 복잡도에 차이가 발생하기 때문이다. 적용 사례에서 본 것처럼 일반적으로 평균 복잡도가 실제 복잡도보다 크게 책정된 이유는 무엇일까? 그 이유는 크게 두 가지로 설명될 수 있다.

첫째, 평균 복잡도는 과거 개발된 소프트웨어의 기능 점수 산정 결과를 통계 분석하여 만들어진 것이다. 평균 복잡도란 여러 소프트웨어 개발 경험을 토대로 만들어진 평균값이다. 만약 순수 수학적 개념의 평균값을 복잡도가 높은 소프트웨어 개발에 적용한다면 프로젝트에 필요한 예산을 충분히 확보하지 못할 수도 있다.

둘째, 간이법은 사업 초기에 예산 수립을 목적으로 적용되는 기능 점수 산정 방법이다. 만약 기능 점수를 기반으로 책정된 예산의 규모가 실제 프로젝트 진행에 필요한 예산보다 적을 경우, 프로젝트 진행에 차질이 빚어질 수 있다. 개발 조직은 프로젝트를 안정적으로 진행하기 위해 어느 정도 여유 있는 예산을 확보하기를 기대할 것이다. 인터넷 쇼핑몰의 경우 데이터 기능과 트랜잭션 기능에 적용할 수 있는 복잡도의 크기가 일반적인 복잡도의 크기보다 다소 낮아 정규법에 의한 규모 산정이 낮게 나왔다고 볼 수 있다.



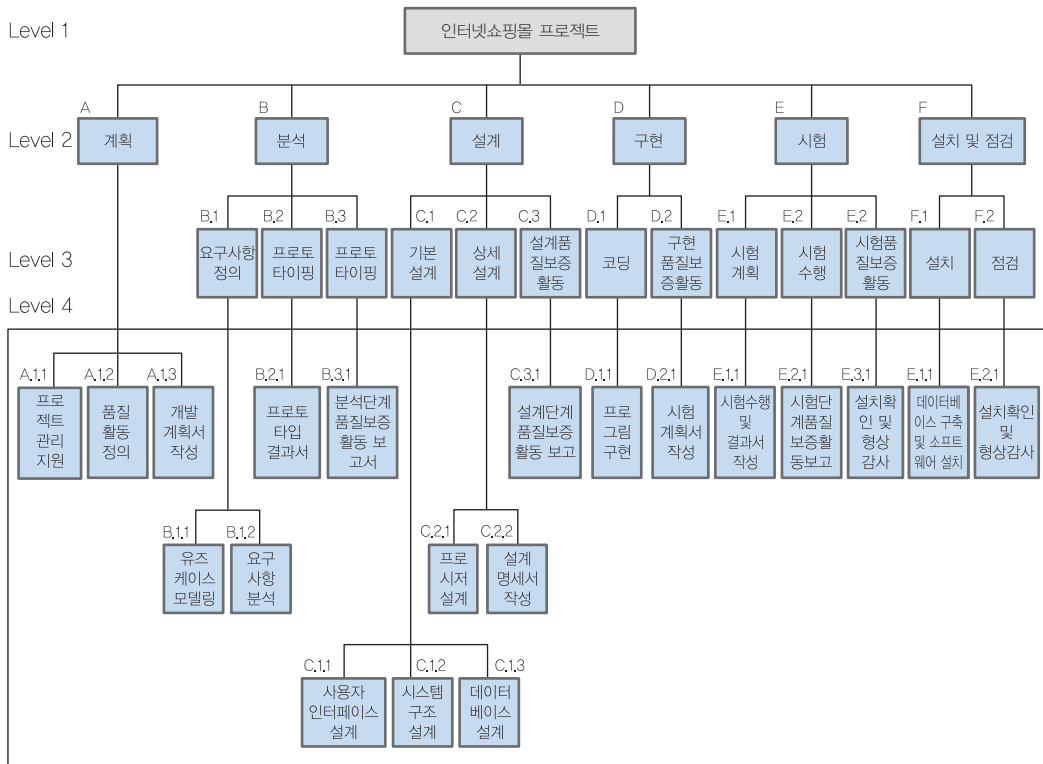
A p p e n d i x

J

인터넷 쇼핑몰 프로젝트 일정 관리 (PERT/CPM 적용)

J / 인터넷 쇼핑몰 프로젝트 일정 관리(PERT/CPM 적용)

인터넷 쇼핑몰 프로젝트의 범위 관리에서 작성된 WBS를 바탕으로 PERT/CPM 기법을 이용하여 일정을 관리하는 과정을 살펴보자.



:: 그림 1 | 인터넷 쇼핑몰 프로젝트의 WBS

그림 1은 범위 관리에서 도출된 인터넷 쇼핑몰 프로젝트의 WBS이다. 하위 레벨의 활동은 상위 레벨의 활동을 세분화하여 나타낸 것이다. 우선 WBS를 작성하여 프로젝트를 성공적으로 수행하기 위해 요구되는 활동을 알아낸다. 하지만 WBS는 단순히 프로젝트 범위를 보여주는 기법이며, 프로젝트의 인도물(또는 산출물)을 생성하기 위해 수행해야 하는 활동만이 나타날 뿐, 이러한 활동을 수행하는 데 필요한 예상 시간이나, 각 활동 간의 선-후행 관계 및 어떠한 활동이 상대적으로 더 중요하고 여유 시간을 갖고 작업을 진행할 수 있는지 등은 알 수 없다.

이러한 시간 관리 문제를 해결하기 위해 앞에서 언급했던 PERT/CPM을 사용하여 프로젝

트를 분석해 보도록 하자. 인터넷 쇼핑몰 프로젝트의 경우 프로젝트의 소요 기간이 비교적 확실하고 반복 사업을 통한 경험이 있을 가능성도 많이 때문에 소요 기간을 프로젝트 팀에서 예측했다고 가정하고 분석을 진행하도록 한다.

각 활동의 소요 기간을 산정한 다음에는 활동 간의 선-후행 관계를 알아내야 한다. 이 과정은 매우 중요한 과정이므로 신중하게 결정하도록 한다. 만일 이 관계가 잘못될 경우 뒤에서 수행하는 분석이 의미가 없어지기 때문이다.

표 1은 앞에 있는 WBS의 작업 패키지의 예상 수행 기간 및 선행 활동을 나타낸 것이다.

● 표 1 | 인터넷 쇼핑몰 프로젝트의 작업 패키지

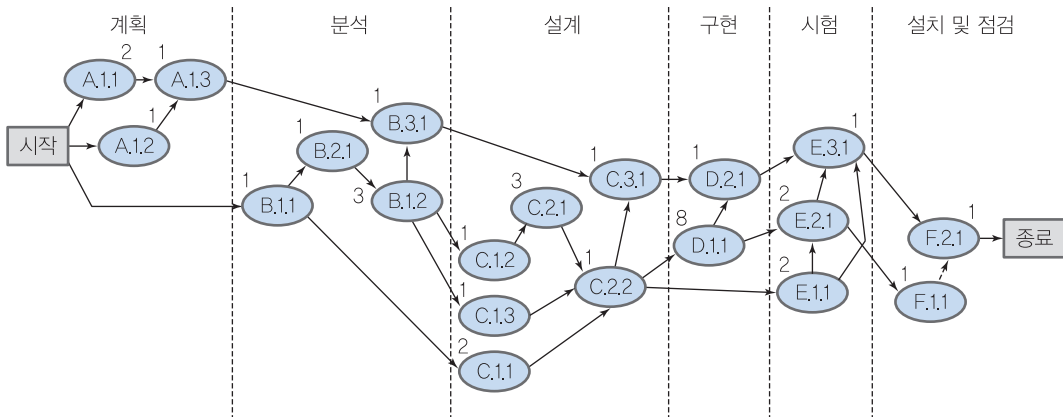
활동	활동 설명	기간(주)	선행 활동
A,1,1	프로젝트 관리 지원 계획	2	none
A,1,2	품질 활동 계획	1	none
A,1,3	개발 계획서 작성	1	A,1,1, A,1,2
B,1,1	유스케이스 모델링	1	none
B,1,2	요구사항 분석(객체지향 분석)	3	B,2,1
B,2,1	프로토타입 구축 및 검증	1	B,1,1
B,3,1	분석 단계 품질 보증 활동 보고	1	A,1,3, B,1,2
C,1,1	사용자 인터페이스 설계	2	B,1,1
C,1,2	시스템 구조 설계	1	B,1,2
C,1,3	데이터베이스 설계	1	B,1,2
C,2,1	프로시저 설계	3	C,1,2
C,2,2	설계 명세서 작성	1	C,1,1, C,1,3, C,2,1
C,3,1	설계 단계 품질 보증 활동 보고	1	B,3,1, C,2,2
D,1,1	프로그램 구현	8	C,2,2
D,2,1	구현 단계 품질 보증 활동 보고	1	C,3,1, D,1,1
E,1,1	시험 계획(단위, 통합, 시스템, 인수)	2	C,2,2
E,2,1	시험 수행(단위, 통합, 시스템, 인수)	2	D,1,1, E,1,1
E,3,1	시험 단계 품질 보증 활동 보고	1	D,2,1, E,1,1, E,2,1
F,1,1	데이터베이스 구축 및 소프트웨어 설치	1	E,2,1
F,2,1	설치 확인 및 형상 감사	1	E,3,1, F,1,1

각 활동을 순차적으로 수행한다면 프로젝트 전체의 수행 기간(소요 시간 추정치의 합)은 35주가 걸릴 것이다. 그러나 활동의 선후 관계를 살펴보면 몇몇 활동들은 동시에 진행할 수 있어

순차적으로 시행했을 때의 수행 기간보다 프로젝트 수행 기간을 훨씬 단축시킬 수 있다. 위에 나타난 표만으로는 각 활동들 사이의 관계를 한눈에 알아내기 어렵고, 이를 그림으로 쉽게 이해하기 위하여 네트워크 다이어그램(network diagram)을 작성한다.

실제로 활동 목록을 바탕으로 네트워크 다이어그램을 작성하면 표 1에서보다 각 활동 간의 관계를 명확하게 알 수 있다. 활동 목록을 작성할 때 주의해야 할 점은 각 활동 간의 선행 활동을 결정하는 일이다. 처음에 구한 선행 활동에 따라서 네트워크 다이어그램의 가치가 결정된다고도 볼 수 있다. 선행 활동이 타당하지 않거나 잘못 설정되어 있다면 그것을 바탕으로 그린 네트워크 다이어그램 역시 타당하지 않거나 잘못된 정보를 제공해 주기 때문이다. 결국 네트워크 다이어그램을 작성하기 전에 프로젝트 매니저와 팀원들 사이에 충분한 토론을 통하여 합리적인 선행 관계를 설정할 필요가 있다.

이제 앞의 표를 바탕으로 네트워크 다이어그램을 작성하여 보자. 네트워크 다이어그램을 작성하게 되면 전체 프로젝트를 한눈에 알아볼 수 있고 각각의 활동들의 관계를 파악하기가 훨씬 수월하다는 장점이 있다. 앞의 표를 바탕으로 네트워크 다이어그램을 작성하면 그림 2와 같다.



::그림 2 | 인터넷 쇼핑몰 네트워크 다이어그램

위의 다이어그램에서 동그라미는 각 활동을 나타내며, 동그라미 위에 써 있는 숫자는 각 활동의 예상 수행 기간이고, 각 활동 간에 그려진 화살표는 활동들의 선행 관계를 나타낸다. 이렇게 네트워크 다이어그램을 작성함으로써 각 활동 간의 관계를 명확하게 알아볼 수 있다.

이제 위에서 작성한 네트워크 다이어그램에 나타난 각각의 활동들에 대해서 프로젝트에 영향을 주지 않는 선에서 활동 수행 시간을 조절할 수 있는 여유 시간을 구해 보자. 여유 시간을 구하기 위해서는 우선 빠른 시작 및 종료일(ES, EF), 늦은 시작 및 종료일(LS, LF)을 구해야 한다.

제일 처음에 작성된 표 1과 네트워크 다이어그램을 바탕으로 각 활동의 빠른 시작일(ES)과 빠른 종료일(EF)을 구한다. 이제 각 활동의 ES와 EF를 구해 보자. 우선 A.1.1이나 A.1.2 또는 B.1.1처럼 선행 활동이 없는 경우 활동을 바로 시작할 수 있기 때문에 ES는 0이 된다. 그리고 EF는 0에서 수행 기간을 더한 값이 된다. A.1.1의 경우 ES는 0, EF는 2가 되고 A.1.2의 경우 ES는 0, EF는 1이 된다.

다음으로 선행 활동이 있는 경우이다. 일반적으로 선행 활동의 EF가 후행 활동의 ES가 된다. 하지만 선행 활동이 여러 개일 경우, 선행 활동의 EF 중에 가장 큰 값이 후행 활동의 ES가 된다. C.2.2의 경우 선행 활동이 C.1.1, C.1.3, C.2.1이고 각각의 EF는 3, 6, 9이다. C.2.2의 ES는 선행 활동의 EF 값 중에 가장 큰 값이 되고 가장 큰 값이 9이므로 C.2.2의 ES는 9이다. 그리고 C.2.2의 EF는 ES 값에서 활동의 수행 기간을 더한 값인 10이 된다.

이러한 방법으로 각각의 활동의 ES와 EF를 구한 값은 표 2와 같다.

::표 2 | 인터넷 쇼핑몰 프로젝트의 작업 패키지(ES, EF)

활동	활동 설명	기간(주)	선행 활동	ES	EF
A.1.1	프로젝트 관리 지원 계획	2	none	0	2
A.1.2	품질 활동 계획	1	none	0	1
A.1.3	개발 계획서 작성	1	A.1.1, A.1.2	2	3
B.1.1	유스케이스 모델링	1	none	0	1
B.1.2	요구사항 분석(객체지향 분석)	3	B.2.1	2	5
B.2.1	프로토타입 구축 및 검증	1	B.1.1	1	2
B.3.1	분석 단계 품질 보증 활동 보고	1	A.1.3, B.1.2	5	6
C.1.1	사용자 인터페이스 설계	2	B.1.1	1	3
C.1.2	시스템 구조 설계	1	B.1.2	5	6
C.1.3	데이터베이스 설계	1	B.1.2	5	6
C.2.1	프로시저 설계	3	C.1.2	6	9
C.2.2	설계 명세서 작성	1	C.1.1, C.1.3, C.2.1	9	10
C.3.1	설계 단계 품질 보증 활동 보고	1	B.3.1, C.2.2	10	11
D.1.1	프로그램 구현	8	C.2.2	10	18
D.2.1	구현 단계 품질 보증 활동 보고	1	C.3.1, D.1.1	18	19
E.1.1	시험 계획(단위, 통합, 시스템, 인수)	2	C.2.2	10	12
E.2.1	시험 수행(단위, 통합, 시스템, 인수)	2	D.1.1, E.1.1	18	20
E.3.1	시험 단계 품질 보증 활동 보고	1	D.2.1, E.1.1, E.2.1	20	21

(계속)

활동	활동 설명	기간(주)	선행 활동	ES	EF
F.1.1	데이터베이스 구축 및 소프트웨어 설치	1	E.2.1	20	21
F.2.1	설치 확인 및 형상 감사	1	E.3.1, F.1.1	21	22

모든 활동에 대하여 ES와 EF를 구하면 이제 전체 프로젝트를 가장 빨리 완료할 수 있는 기간을 알 수 있다. 표 2에 나타난 결과에 따르면 전체 프로젝트를 가장 빨리 완료할 수 있는 기간은 총 22주가 된다. 처음에 구했던 모든 활동을 순차적으로 수행했을 때와 비교하면 많은 시간을 절약할 수 있음을 알 수 있다.

이제 각 활동의 LF와 LS를 구하는 방법을 몇 가지 예를 들어서 알아보자. 이번에는 네트워크 다이어그램과 표 1 그리고 표 2에서 구했던 EF 값을 이용한다. 표 1이나 네트워크 다이어그램을 바탕으로 각 활동의 후행 활동을 구한다. 종료 시점부터 역으로 올라가며 각 활동의 LF, LS를 구하는데, 앞에서 구했던 ES, EF와는 달리 LF를 먼저 구하고 LS를 구한다. ES, EF 때는 선행 활동이 없었던 활동부터 계산하기 시작했다면 LF, LS는 후행활동이 없는 활동부터 계산하도록 한다. 후행 활동이 없는 F.2.1의 경우 가장 늦은 완료 시점인 LF는 앞에서 구했던 F.2.1의 EF 값인 22이고, 가장 늦은 시작 시점인 LS는 수행 기간 1을 뺀 21이 된다.

다음으로 후행 활동이 있는 경우에 어떻게 LF와 LS를 구하는지 알아보자. 일반적으로 후행 활동의 LS 값이 선행 활동의 LF 값이 된다. 후행 활동이 여러 개 있을 경우 후행 활동의 LS 값 중 가장 작은 값이 선행 활동의 LF 값이 된다. 예를 들어서 알아보자. 위의 활동 중에 E.1.1의 경우 후행 활동은 E.2.1, E.3.1과 F.1.1이다. 후행 활동의 LS는 각각 18, 20, 20이다. 이들 중에 가장 작은 값이 E.1.1의 LF가 되고 LF 값에서 활동의 수행 기간을 뺀 값이 LS가 되므로 E.1.1의 경우 LF는 18, LS는 16이 된다. 이러한 방법으로 각 활동의 LF, LS를 구하면 표 3과 같다.

::표 3 | 인터넷 쇼핑몰 프로젝트의 작업 패키지(LF, LS)

활동	활동 설명	기간(주)	후행 활동	LF	LS
A.1.1	프로젝트 관리 지원 계획	2	A.1.3	16	14
A.1.2	품질 활동 계획	1	A.1.3	16	15
A.1.3	개발 계획서 작성	1	B.3.1	17	16
B.1.1	유스케이스 모델링	1	B.2.1, C.1.1	1	0
B.1.2	요구사항 분석(객체지향 분석)	3	B.3.1, C.1.2, C.1.3	5	2
B.2.1	프로토타입 구축 및 검증	1	B.1.2	2	1

(계속)

활동	활동 설명	기간(주)	후행 활동	LF	LS
B.3.1	분석 단계 품질 보증 활동 보고	1	C.3.1	18	17
C.1.1	사용자 인터페이스 설계	2	C.2.2	9	7
C.1.2	시스템 구조 설계	1	C.2.1	6	5
C.1.3	데이터베이스 설계	1	C.2.2	9	8
C.2.1	프로시저 설계	3	C.2.2	9	6
C.2.2	설계 명세서 작성	1	C.3.1, D.1.1, E.1.1	10	9
C.3.1	설계 단계 품질 보증 활동 보고	1	D.2.1	19	18
D.1.1	프로그램 구현	8	D.2.1, E.2.1	18	10
D.2.1	구현 단계 품질 보증 활동 보고	1	E.3.1	20	19
E.1.1	시험 계획(단위, 통합, 시스템, 인수)	2	E.2.1, E.3.1, F.1.1	18	16
E.2.1	시험 수행(단위, 통합, 시스템, 인수)	2	E.3.1, F.1.1	20	18
E.3.1	시험 단계 품질 보증 활동 보고	1	F.2.1	21	20
F.1.1	데이터베이스 구축 및 소프트웨어 설치	1	F.2.1	21	20
F.2.1	설치 확인 및 형상 감사	1	none	22	21

이제 각 활동의 여유 시간을 구해 보자. 여유 시간은 각 활동이 전체 프로젝트 수행 기간에 영향을 주지 않는 선에서 해당 활동의 시작 시간을 조정할 수 있는 값이다. 즉, 어떤 활동의 여유 시간이 4주라면 해당 활동의 시작이 예정 시작 시간보다 시작이 늦어진다고 해도 예정 시작 시간으로부터 4주 안에만 시작하면 전체 프로젝트 수행 기간에는 영향을 주지 않게 된다.

여유 시간은 각 활동의 $LF - EF$ 또는 $LS - LF$ 이다. 앞의 두 가지 방법 중에 어느 것을 사용해도 무방하며 두 가지 방법 모두 같은 여유 시간을 구하게 된다. 두 가지 방법 모두 같은 값을 구하는 것을 보이기 위해 표 4에서는 두 가지 방법으로 구한 여유 시간을 모두 표시하도록 한다. 각 활동의 여유 시간을 구한 값은 표 4와 같다.

●표 4 | 인터넷 쇼핑몰 프로젝트의 작업 패키지(ES, EF, LF, LS)

활동	활동 설명	기간(주)	선행 활동	ES	EF	후행 활동	LF	LS	여유 시간 (LS-ES)	여유 시간 (LF-EF)	주경로
A.1.1	프로젝트 관리 지원 계획	2	none	0	2	A.1.3	16	14	14	14	F
A.1.2	품질 활동 계획	1	none	0	1	A.1.3	16	15	15	15	F
A.1.3	개발 계획서 작성	1	A.1.1, A.1.2	2	3	B.3.1	17	16	14	14	F

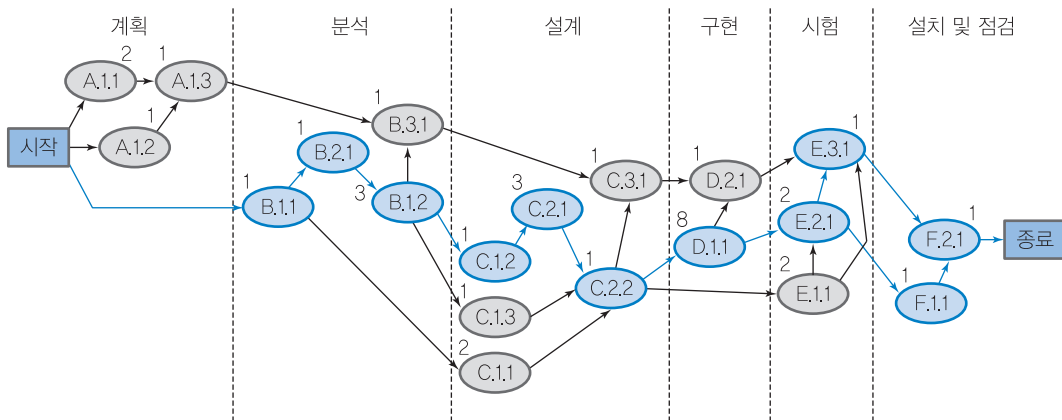
(계속)

활동	활동 설명	기간(주)	선행 활동	ES	EF	후행 활동	LF	LS	여유 시간 (LS-ES)	여유 시간 (LF-EF)	주경로
B.1.1	유스케이스 모델링	1	none	0	1	B.2.1, C.1.1	1	0	0	0	T
B.1.2	요구사항 분석 (객체지향 분석)	3	B.2.1	2	5	B.3.1, C.1.2, C.1.3	5	2	0	0	T
B.2.1	프로토타입 구축 및 검증	1	B.1.1	1	2	B.1.2	2	1	0	0	T
B.3.1	분석 단계 품질 보증 활동 보고	1	A.1.3, B.1.2	5	6	C.3.1	18	17	12	12	F
C.1.1	사용자 인터페이스 설계	2	B.1.1	1	3	C.2.2	9	7	6	6	F
C.1.2	시스템 구조 설계	1	B.1.2	5	6	C.2.1	6	5	0	0	T
C.1.3	데이터베이스 설계	1	B.1.2	5	6	C.2.2	9	8	3	3	F
C.2.1	프로시저 설계	3	C.1.2	6	9	C.2.2	9	6	0	0	T
C.2.2	설계 명세서 작성	1	C.1.1, C.1.3, C.2.1	9	10	C.3.1, D.1.1, E.1.1	10	9	0	0	T
C.3.1	설계 단계 품질 보증 활동 보고	1	B.3.1, C.2.2	10	11	D.2.1	19	18	8	8	F
D.1.1	프로그램 구현	8	C.2.2	10	18	D.2.1, E.2.1	18	10	0	0	T
D.2.1	구현 단계 품질 보증 활동 보고	1	C.3.1, D.1.1	18	19	E.3.1	20	19	1	1	F
E.1.1	시험 계획 (단위, 통합, 시스템, 인수)	2	C.2.2	10	12	E.2.1, E.3.1, F.1.1	18	16	6	6	F
E.2.1	시험 수행 (단위, 통합, 시스템, 인수)	2	D.1.1, E.1.1	18	20	E.3.1, F.1.1	20	18	0	0	T
E.3.1	시험 단계 품질 보증 활동 보고	1	D.2.1, E.1.1, E.2.1	20	21	F.2.1	21	20	0	0	T
F.1.1	데이터베이스 구축 및 소프트웨어 설치	1	E.2.1	20	21	F.2.1	21	20	0	0	T
F.2.1	설치 확인 및 형상 감사	1	E.3.1, F.1.1	21	22	none	22	21	0	0	T

프로젝트 전체의 수행 기간을 효과적으로 줄일 수 있는 방법 중에 하나로 주경로(critical path) 분석이 있다. 주경로는 주활동(critical activity)으로 이루어진 경로이다. 여기서 주활동이

란 여유 시간이 0인 활동이며, 주경로상에 있는 주활동이 예상 수행 시간보다 늦어지게 된다면 프로젝트 전체의 수행 기간이 늦어지는 결과를 초래하는 활동을 말한다. 그러므로 전체 프로젝트의 지연을 억제하기 위해서는 주경로의 지연을 억제하여야 한다. 반대로 주경로상의 활동들이 예정보다 일찍 완료된다면 프로젝트 전체의 수행 기간도 일찍 완료될 수 있을 것이다.

앞의 표 4를 살펴보면 (LS-ES)와 (LF-EF)의 두 가지 방법으로 구한 여유 시간이 모두 같음을 알 수 있다. 그리고 여유 시간이 0인 활동들이 주활동이 된다. 그리고 이러한 활동들이 모인 경로가 구하려고 했던 주경로가 된다. 앞의 표 4를 바탕으로 네트워크 다이어그램에 주경로를 나타내면 그림 3과 같다.



::그림 3 | 인터넷 쇼핑몰 네트워크 다이어그램(주경로 포함)

위의 그림에 파란색으로 표시된 활동과 화살표가 각각 주활동과 주경로다. 주경로상에 있는 주활동들은 여유 시간이 0이기 때문에 지연이 발생하면 프로젝트 전체에 지연을 발생시킨다. 그러므로 프로젝트의 지연을 막으려면 주경로상에 있는 주활동의 지연을 막아야 하고 반대로 프로젝트의 수행 시간을 단축시키려면 주활동의 수행 시간을 단축시키면 된다.

위의 네트워크 다이어그램을 살펴보면 두 가지 주경로가 나타난다. 각각을 살펴보면 다음과 같다. (B.1.1 - B.2.1 - B.1.2 - C.1.2 - C.2.1 - C.2.2 - D.1.1 - E.2.1 - E.3.1 - F.2.1)과 (B.1.1 - B.2.1 - B.1.2 - C.1.2 - C.2.1 - C.2.2 - D.1.1 - E.2.1 - F.1.1 - F.2.1)이다. 이렇게 두 개의 주경로를 갖게 되면 한 개의 주경로를 가지고 있을 때보다 주활동의 숫자가 늘어나기 때문에 주활동을 관리하는 데 있어서 더 많은 노력이 필요하다. 만일 한쪽 주경로에서 지연이 발생하면 다른 쪽 주경로가 제시간에 완료되었다 하더라도 프로젝트 전체의 수행 시간이 지연되기 때문이다.

한 프로젝트에 대해 주경로가 여러 개 나타날 경우, 주경로가 한 개일 때보다 프로젝트 관리 위험이 증가함에도 불구하고 주경로 분석은 이러한 위험 증가 사실을 알려 주지 않고 프로젝트 전체의 수행 기간을 알려줄 뿐이다. 결국 주경로 분석은 프로젝트 일정 관리에 적합하지만 위험 관리 측면에서는 한계를 가지고 있다고 할 수 있다.

이러한 주경로 분석이 완료되면 주경로가 표시된 네트워크 다이어그램 또는 간트 차트를 프로젝트에 참여하는 인원이 쉽게 볼 수 있는 장소에 게시하여 일정을 파악할 수 있도록 한다. 주경로 분석이 완료된 네트워크 다이어그램(또는 간트 차트)을 통해 프로젝트 각 업무의 시작과 끝을 알 수 있으며, 프로젝트 전체 일정을 한눈에 알아볼 수 있다. 또한 프로젝트의 주활동과 주경로를 바로 알아볼 수 있어 현재 수행 중인 작업이 프로젝트의 일정에 어떠한 영향을 미치는지 파악할 수 있다. 이런 정보를 통해 프로젝트 관리자는 프로젝트의 일정 관리를 좀 더 효율적으로 수행할 수 있고, 프로젝트 팀원들도 자신이 수행하는 업무의 중요성을 알고 책임감이 들 수 있을 것이다.



A p p e n d i x

K

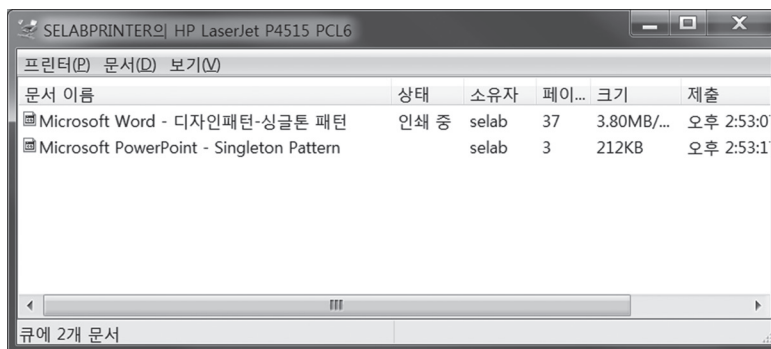
디자인 패턴의 실제 예

1. Singleton Pattern
2. Façade Pattern
3. Strategy Pattern
4. Factory Method Pattern
5. Adapter Pattern

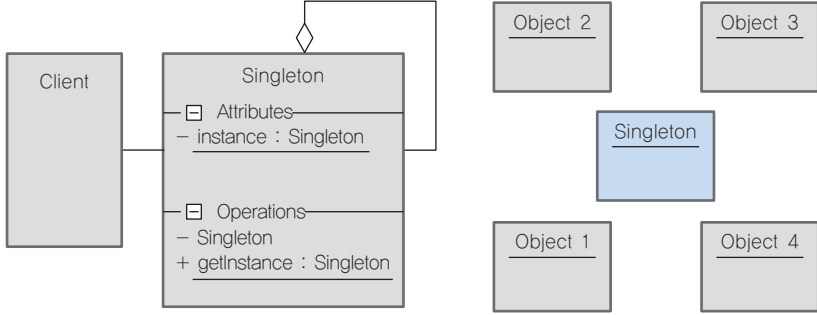
1 Singleton Pattern

Singleton Pattern은 객체의 생성에 관련된 패턴으로서 특정 클래스의 인스턴스가 오직 하나임을 보장하고(클래스의 객체를 한 객체로 제한), 이 인스턴스에 접근할 수 있는 방법을 제공한다.

앞서 설명한 인쇄 관리 프로그램을 다시 한 번 예로 든다. 인쇄 요청은 여러 프로그램에서 동시에 올 수 있고, 한 번에 한 페이지에서부터 수백 페이지까지 다양한 요청이 올 수 있다. 만일 여러 프로그램에서 서로 다른 여러 장의 문서 또는 이미지를 프린터기로 인쇄를 요청하면 인쇄물이 섞이는 일이 발생할 수도 있을 것이다. 때문에 하나의 인쇄 관리 프로그램을 통해서 프린터기로 인쇄할 문서의 페이지를 차례차례 보내게 된다.



앞의 프린트 관리 객체처럼 클라이언트의 요청을 유일하게 존재하는 인스턴스로 접근을 통제해야 할 때 Singleton Pattern을 사용한다. 다시 말해, Singleton Pattern은 어떤 클래스의 객체가 시스템에서 유일한 단일 인스턴스가 되도록 보장해 주고 이 인스턴스에 접근할 수 있는 방법을 제공해 주는 패턴이다.

이름	Singleton Pattern
분류	목적 : 생성 범위 : 객체
의도	클래스의 인스턴스는 오직 하나임을 보장하며 singleton 인스턴스에 접근할 수 있는 방법을 제공한다.
구조	 <ul style="list-style-type: none"> Singleton은 자기 자신의 클래스 타입의 static 변수를 갖는다. 또한 getInstance() 메소드에 의해 오직 한 개만 생성하여 제공한다(다음 소스 코드 참조). 때문에 자기 자신과 연관(UML의 association)을 갖는 구조적 특징을 보인다. 구조도의 오른쪽 그림은 프로그램에 생성되는 객체를 표현한 그림이다. UML의 객체 다이어그램으로 표현할 경우 위의 오른쪽 그림처럼 Client 클래스는 여러 개의 객체가 생성되어 존재하지만 Singleton 클래스는 하나의 Singleton 객체만을 생성하는 구조를 갖는다.
클래스 설명	Singleton : 생성되는 객체, 인스턴스를 반환하는 클래스 레벨(정적) 메소드를 정의한다. 클래스 레벨의 getInstance 메소드에 의해 객체가 반환된다.

✓ 샘플 코드

Singleton Pattern을 만드는 코드는 간단하다.

- (1) private 접근 제한자를 갖는 멤버변수로 자기 자신의 클래스 인스턴스를 갖고
- (2) 생성된 인스턴스를 반환할 수 있는 static 메소드를 정의(예를 들어, getInstance())하고
- (3) private 생성자를 지정하여 외부에서 위 (2) 항목의 static 메소드를 이용하지 않고서는 절대로 인스턴스를 생성하지 못하게 한다.

Singleton Pattern 기본 코드

```
public class Singleton {

    private static Singleton instance;

    private Singleton() {
        // 기본 생성자가 private이기 때문에 외부에서 인스턴스를 생성할 수 없다.
    }

    // Singleton 객체는 getInstance() 메소드를 통해서만 객체가 생성된다.
    public static Singleton getInstance() {
        if(instance==null) {
            instance = new Singleton();
        }

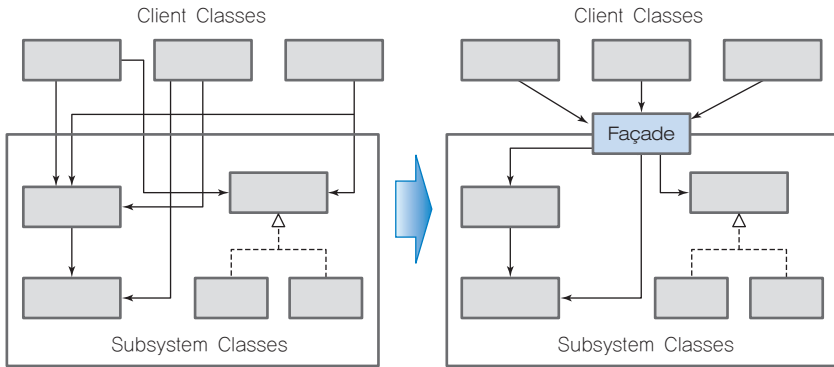
        return instance;
    }
}
```

Singleton Pattern은 단순한 구조를 갖지만, 동일한 자원이나 데이터를 처리하는 객체가 불필요하게 여러 개 만들어질 필요가 없는 경우에 주로 사용되며, 순차적으로 프로그램 내의 환경 설정 관리 클래스, 네트워크 프로그램에서 연결 처리나 thread 관리를 pool 형태로 해야 할 경우, 게임에서 동일한 캐릭터의 개수를 관리해야 할 경우 등에서 유용하다.

2 Façade Pattern

Façade란 건물에서 거리 또는 공간(정원, 광장 등) 쪽을 향하고 있는 건물의 앞쪽을 말하며 건물의 주된 정문으로 현관을 포함하고 있는 부분을 말한다. 건물의 정문은 사람들이 가장 많이 왕래하는 곳이기 때문에 큰 건물의 경우에는 ‘안내소’가 위치한다. 호텔이나 극장의 로비들도 대부분 정문 쪽에 위치하기 때문에 건물의 안에서든 밖에 있을 때든 안내를 받아야 할 경우나 요금을 계산해야 할 경우 자연스럽게 건물의 정문 쪽을 찾게 된다. Façade Pattern도 건물의 정문에 있는 안내소처럼 개발자가 사용해야 하는 서브시스템의 가장 앞쪽에 위치하면서 하위 시스템에 있는 객체들을 사용할 수 있도록 하는 역할을 한다. 다시 말해, Façade

Pattern은 시스템의 복잡성을 줄이기 위해 서브시스템을 구조화하고 서브시스템으로의 접근을 하나의 Façade 객체로 제공하는 패턴이다. 때문에 GoF는 Façade Pattern을 목적상 구조적 패턴으로 범위상 객체와 관련된 패턴으로 분류하고 있다.

이름	Façade Pattern
분류	목적 : 구조 범위 : 객체
의도	하나의 인터페이스를 통해 복잡한 하위 시스템에 접근하도록 한다.
구조	 <p>Façade는 시스템 간 복잡한 연관 관계가 있을 경우 두 시스템 사이에 위치하여 복잡성을 낮추는 역할을 한다.</p>
클래스 설명	<ul style="list-style-type: none"> • Façade: 복잡한 하위 시스템에 대한 인터페이스를 제공하는 역할을 한다. 위 그림에서 subsystem을 이용할 경우 모든 연결은 Façade를 통해서 이루어진다. • Subsystem: 하나 이상의 클래스들로 구성되어 있으며, 클라이언트가 원하는 서비스를 수행하는 클래스들이다.

‘Façade Pattern 적용 전’과 ‘Façade Pattern 적용 후’를 보면 첫 번째 그림은 클래스들 간 연관이 매우 복잡해 보인다. 이런 복잡한 연관은 클라이언트들과 하위 시스템 간 의존 관계가 너무 많아 결합도(coupling)가 높은 상태이므로 이를 감소시킬 필요가 있음을 의미한다. ‘Façade Pattern 적용 전’의 클래스 간 높은 의존성은 ‘Façade Pattern 적용 후’에서 보듯 클라이언트와 서브시스템 사이에 Façade 객체를 추가하여 클라이언트가 다루어야 할 객체의 수를 줄이고 있다. 결과적으로 서브시스템과 클라이언트 코드 간의 의존도와 결합도를 낮출 수 있다. 즉, Façade Pattern은 시스템 간 의존 관계를 낮추고자 할 때 유용하다. 수십 개의 각종 서비스들이 통합되고 엄청난 규모의 모듈들이 서로 연동되어야 하는 대규모 프로젝트에서 복잡성과 결합도를 줄이는 가장 효율적인 방법이다. Façade Pattern은 가장 단순한 패턴이기 때문에 다양한 스킬 레벨을 가진 수십, 수백 명의 개발자들이 모두 함께 통일된 모습으

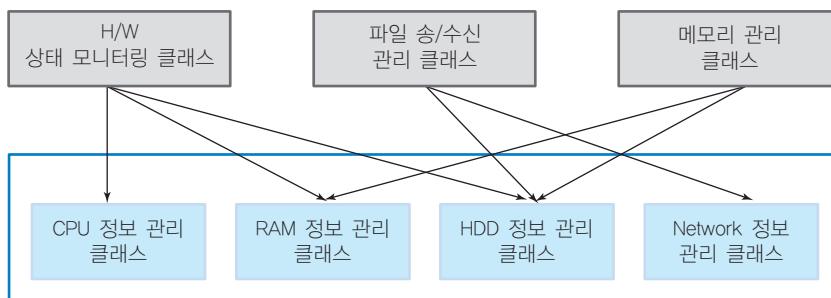
로 개발하기 가장 쉬운 패턴이기도 하다.

마지막으로 Façade Pattern의 적용이 필요한 상황을 정리해 보면 다음과 같다.

- 복잡한 서브시스템에 대한 단순한 인터페이스 제공이 필요할 때
- 클라이언트와 구현 클래스 간에 너무 많은 의존성이 존재하여 클라이언트와 다른 서브시스템 간의 결합도를 줄일 필요가 있을 때
- 빌딩 블록(building block) 아키텍처나 컴포넌트 기반 개발(component based development), Service Oriented Architecture 등의 경우와 같이 서로의 내부 구조를 감추고 블랙박스로 이해해야 할 때
- 서브시스템들이 계층화를 이루어 각 서브시스템의 계층별 안내소인 접근점(Façade 객체)을 제공하려 할 때

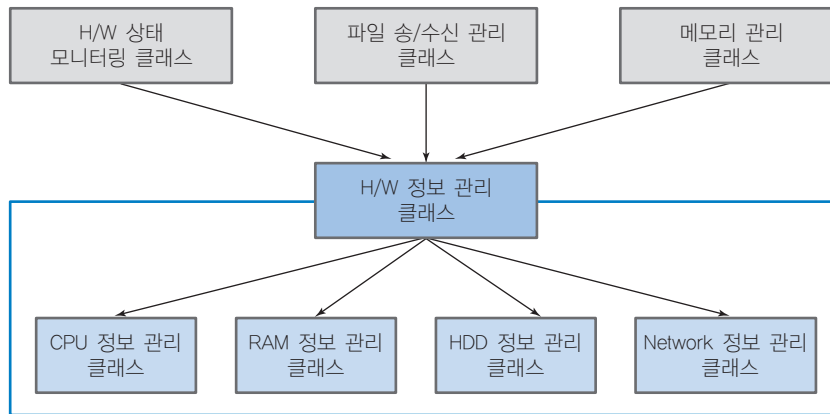
다음은 인터넷을 이용하다 많이 사용하게 되는 파일 송수신 프로그램의 Façade 사용 예이다. 인터넷이 보급되면서 다양한 종류의 파일 송수신 프로그램이 만들어져 사용되고 있다. 파일 송수신 프로그램들은 단순히 파일을 인터넷을 통해 전송하거나 수신하고 파일을 저장하는 기능만 제공하지 않는다. 사용자의 네트워크 상태에 따라 전송 속도를 조절하기도 하고 하드디스크 용량을 체크하여 파일을 수신할 수 있는지 점검 기능을 제공하기도 한다. 부가적으로 전송 및 수신되는 속도나 수신된 파일 용량을 실시간으로 그래픽컬하게 보여주기도 한다.

만일 위와 같은 기능의 프로그램을 만들기 위해 설계를 한다면 초보 개발자들은 그림 1과 같은 구조로 설계한다.



::그림 1 | 높은 결합도 구조

간단히 설명하자면 하드웨어 정보가 필요한 시점에서 해당 하드웨어 정보를 제공해 주는 클래스의 인스턴스를 생성하여 사용하기만 하면 된다. 이러한 구조는 그림 속의 선들이 얽혀 있어 복잡해 보이는 만큼 결합도가 높아진다. 이러한 복잡도를 낮추는 방법을 제공하는 패턴이 Façade Pattern이다.



::그림 2 | Façade Pattern 구조

그림 2와 같이 그림에 보이는 구조적 단순함은 소스 코드로도 확인할 수 있다. 다음은 ‘H/W 상태 모니터링 클래스’가 CPU, RAM, HDD 정보를 확인하기 위한 코드로 Façade Pattern 적용 전후의 코드이다.

Façade Pattern을 사용하지 않을 경우 ‘H/W 상태 모니터링 클래스’는 하드웨어 정보를 그래프 방식으로 출력하기 위해 ‘하드디스크 정보 관리 클래스(HDDInfoManager)’, ‘CPU 정보 관리 클래스(CPUInfoManager)’, ‘RAM 정보 관리 클래스(RAMInfoManager)’를 필요할 때마다 각각 호출하여 사용하는 구조이다. 이렇게 사용할 경우 ‘하드디스크 정보 관리 클래스’, ‘CPU 정보 관리 클래스’, ‘RAM 정보 관리 클래스’ 중에서 하나라도 변경이 이루어지면 즉각적으로 아래의 하드웨어 상태를 그래프로 출력하는 메소드(displayHWstateGraph)에 영향을 미치게 된다.

그러나 Façade Pattern을 적용하여 구조를 개선하면 ‘H/W 상태 모니터링 클래스’는 CPU, HDD, RAM 정보를 제공하는 ‘H/W 정보 관리 클래스(HWInfoManager)’만의 영향을 받게 된다. 즉, 단순한 인터페이스(H/W 정보 관리 클래스)를 사용해 결합도가 낮아지는 장점을 즉각적으로 볼 수 있다.

Façade를 사용하지 않은 구조의 코드

H/W 상태 모니터링 클래스 내 H/W 상태 정보를 그래프로 표출하는 메소드

```
public void displayHwStateGraph() {
    // 현재 HDD 정보는 HDDInfoManager를 이용
    HDDInfo hddInfo = HDDInfoManager.getInstance().getHDDInfo();
    // 현재 CPU 정보는 CPUInfoManager를 이용
    CPUInfo cpuInfo = CPUInfoManager.getInstance().getCPUInfo();
    // 현재 RAM 정보는 RAMInfoManager를 이용
    → RAMInfo ramInfo = RAMInfoManager.getInstance().getRamInfo();
    // HDD 정보 그리픽컬하게 전달
    DrawHWInfo("hdd", hddInfo);
    DrawHWInfo("cpu", cpuInfo);
    DrawHWInfo("ram", ramInfo);
}
```

[코드 설명]

위의 “RAMInfoManager.getInstance().getRamInfo()” API를 만든 개발자가 ‘getRamInfo()’ 메소드를 이용해 지난 시간 Ram의 상태 정보를 알 수 있도록 파라미터를 추가하여 재개발해야만 하는 일이 발생했다고 가정해 보자.

- 변경 전: getRamInfo()
- 변경 후: “getRamInfo(int_pastime)”

이런 경우 getRamInfo()를 사용한 개발자들은 해당 코드를 모두 getRamInfo(0);로 변경하는 작업이 필요할 것이다.

Façade가 사용된 구조의 코드

H/W 상태 모니터링 클래스 내 H/W 상태 정보를 그래프로 표출하는 메소드

Façade를 적용하면 API 사용자는 HDD 정보, CPU 정보, RAM 정보를 “HWInfoManager”를 통해서 알 수 있기 때문에 하드웨어 유형에 따라 제공 클래스를 찾아야 하는 번거로움을 해결할 수 있다.

```
public void displayHwStateGraph() {
    // H/W 정보는 Façade Pattern을 적용하여 만들어진 HWInfoManager를 이용
    HDDInfo hddInfo = HWInfoManager.getInstance().getHDDInfo();
    CPUInfo cpuInfo = HWInfoManager.getInstance().getCPUInfo();
    → RAMInfo ramInfo = HWInfoManager.getInstance().getRamInfo();

    // HDD 정보 그리픽컬하게 전달
    DrawHWInfo("hdd", hddInfo);
    DrawHWInfo("cpu", cpuInfo);
    DrawHWInfo("ram", ramInfo);
}
```

Façade 클래스

// Façade Pattern을 적용하여 만들어진 'H/W 정보 관리 클래스'이다.

```
public class HWInfoManager {
    private static HWInfoManager instance;

    public static HWInfoManager getInstance() {
        if(instance==null) {
            instance = new HWInfoManager( );
        }

        return instance;
    }

    // CPU 정보 반환
    public CPUInfo getCPUInfo() {
        return CPUInfoManager.getInstance().getCPUInfo();
    }

    // HDD 정보 반환
    public HDDInfo getHDDInfo() {
        return HDDInfoManager.getInstance().getHDDInfo();
    }

    // RAM 정보 반환
    public RAMInfo getRamInfo() {
        → return RAMInfoManager.getInstance().getRamInfo();
    }
}
```

[코드 설명]

위의 “RAMInfoManager.getInstance().getRamInfo()” API를 만든 개발자가 ‘getRamInfo()’ 메소드를 이용해 지난 시간 Ram의 상태 정보를 알 수 있도록 파라미터를 추가하여 재개발해야만 하는 일이 발생했다고 가정해 보자.

- 변경 전: getRamInfo()
- 변경 후: “getRamInfo(int_pastime)”

이럴 경우 API 제공자는 Façade 클래스인 HWInfoManager 내 getRamInfo() 메소드 내 코드를 “RAMInfoManager.getInstance().getRamInfo(0)”로 수정하면 사용자들에게는 이전과 동일한 API를 제공할 수 있기 때문에 실제 RAM 정보를 사용하는 ‘HWInfoManager.getInstance().getRamInfo();’ 코드에 영향을 주지 않는다.

3 Strategy Pattern

Strategy Pattern은 다양한 알고리즘이 존재할 때 이들 각각을 하나의 클래스로 캡슐화하여 알고리즘을 대체가 가능하도록 한다. 이를 통해 클라이언트에 영향을 주지 않고 다양한 알고리즘으로 변형할 수 있어 알고리즘을 바꾸더라도 클라이언트는 어떤 변경도 할 필요가 없다. Strategy Pattern을 GoF의 분류적인 관점에서 다시 설명하면 알고리즘을 담당하는 각각의 클래스를 만들어 책임을 분산하기 위한 목적으로 만든 행위 패턴이고, 각각의 알고리즘을 필요한 시점(런타임 시)에서 동적으로 변경하여 사용할 수 있기 때문에 범위적인 측면에서는 객체 패턴이다. Strategy Pattern은 앞에서 설명한 다형성과 동적 바인딩의 개념이 적용된 패턴이라 할 수 있다.

Strategy Pattern이 유용하게 사용되는 경우는 다음과 같다.

- 행위들이 조금씩 다를 뿐 개념적으로 관련된 많은 클래스들이 존재하는 경우, 각각의 서로 다른 행위별로 클래스를 작성한다.
- 알고리즘의 변형이 필요한 경우에 사용할 수 있다. 저장 공간과 처리 속도 간의 절충에 따라 서로 다른 알고리즘을 사용할 수 있다.
- 많은 행위를 정의하기 위해 클래스 안에 복잡한 다중 조건문을 사용해야 하는 경우 이런 선택문보다는 Strategy 클래스로 만드는 것이 바람직하다.
- 어떤 알고리즘이 클라이언트가 알아서는 안 될 데이터를 사용하거나 알고리즘에 종속된 복잡한 자료 구조를 사용할 때 유용하다. 이 경우 Strategy Pattern은 클라이언트에게 알고리즘이 사용하는 데이터나 자료 구조를 숨겨 주는 역할을 한다.

이름	Strategy
분류	목적 : 행위 범위 : 객체
의도	알고리즘군이 존재할 경우 각각의 알고리즘을 별도의 클래스로 캡슐화하고 이들을 상호 교환 가능한 것으로 정의한다. Strategy Pattern은 클라이언트에 영향을 주지 않고 독립적으로 알고리즘을 다양하게 변경할 수 있게 한다.
구조	<pre> classDiagram class Client class Context { +Operations } class Strategy { +Operations +AlgorithmInterface: void } class ConcreteStrategyA { +Operations +AlgorithmInterface: void } class ConcreteStrategyB { +Operations +AlgorithmInterface: void } class ConcreteStrategyC { +Operations +AlgorithmInterface: void } Client ..> Context Context o--> Strategy Strategy < .. ConcreteStrategyA Strategy < .. ConcreteStrategyB Strategy < .. ConcreteStrategyC </pre> <p>Strategy Pattern은 알고리즘의 접근을 허용하는 ‘Strategy 클래스’, 실제 알고리즘이 구현된 ‘ConcreteStrategy 클래스’, 클라이언트들이 알고리즘을 쉽게 사용할 수 있도록 ‘Context 클래스’로 구성한다.</p>
클래스 설명	<ul style="list-style-type: none"> • Strategy: 알고리즘으로의 접근을 허용하는 인터페이스이다. ConcreteStrategyA, ConcreteStrategyB, ConcreteStrategyC: Strategy 인터페이스를 따라 특정 알고리즘을 구현한다. • Context: Strategy 인터페이스를 통해 알고리즘을 사용한다.

✓ 샘플 코드

Strategy Pattern 예(JAVA)

```

/*
 * #1 SortStrategy.java
 * Strategy 인터페이스
 */
package StrategyPattern;

import java.util.ArrayList;

public interface SortStrategy {

    public void sort(ArrayList<Integer> dataList);

}

/*
 * #2 BubbleSort.java
 * Strategy 구상 클래스
 */
package StrategyPattern;

import java.util.ArrayList;

public class BubbleSort implements SortStrategy {

    @Override
    public void sort(ArrayList<Integer> dataList) {
        // TODO Auto-generated method stub
        System.out.println("Run Bubble Sort");
    }

}

/*
 * #3 QuickSort.java
 * Strategy 구상 클래스
 */
package StrategyPattern;

import java.util.ArrayList;

public class QuickSort implements SortStrategy {

    @Override
    public void sort(ArrayList<Integer> dataList) {

```



```

        // TODO Auto-generated method stub
        System.out.println("Run Quick Sort");
    }
}

/*
 * #4 Context.java
 * Strategy 인터페이스를 이용해 알고리즘을 사용하는 클래스
 */
package StrategyPattern;
import java.util.ArrayList;

public class Context {

    private SortStrategy srtStrategy;

    public void setContext(SortStrategy srtStrategy) {
        this.srtStrategy = srtStrategy;
    }

    public void sortExcute(ArrayList<Integer> arrayList) {

        this.srtStrategy.sort(arrayList);
    }
}

/*
 * #5 StrategyPatternTest.java
 * Strategy 패턴 사용 클라이언트 프로그램
 */
package StrategyPattern;

import java.util.ArrayList;
import java.util.Random;

public class StrategyPatternTest {
    /**
     * @param args
     */
    public static void makeNum(ArrayList<Integer> arrayList) {
        Random random = new Random( );

        for(int i=0; i< 10; i++) {
            arrayList.add(random.nextInt(1000));
        }
    }
}

```

```

    }
    public static void main(String[] args) {
        // TODO Auto-generated method stub
        ArrayList<Integer> arrayList= new ArrayList<Integer>( );

        StrategyPatternTest.makeNum(arrayList);

        Context context = new Context( );

        context.setContext(new BubbleSort( ));
        context.sortExcute(arrayList);

        context.setContext(new QuickSort( ));
        context.sortExcute(arrayList);

    }
}

```

실행 결과

```

Problems | @ Javadoc | Declaration | Console
<terminated> StrategyPatternTest [Java Application] C:\Program Files\Java\jre7\bin\javaw.exe
Run Bubble Sort
Run Quick Sort

```

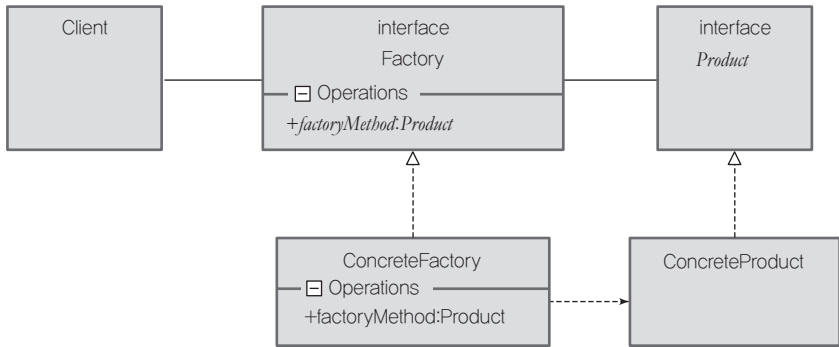
4 Factory Method Pattern

객체를 생성하기 위해 일정한 절차가 필요하거나 객체를 생성하는 시점이 불명확할 경우 객체를 생성하는 메소드를 이용할 수 있다. 예를 들어, 폴더 안에 있는 파일을 사용자가 선택하면 선택한 파일을 사용자가 원하는 시점에 읽을 수 있는 응용 프로그램을 만든다고 가정하자. 폴더 안에는 문서 파일, 그림 파일, 동영상 파일 등 다양한 종류의 파일이 있을 수 있고 파일 유형의 수만큼 연결해야 하는 프로그램도 다양함을 알 수 있다.

사용자가 파일을 선택하면 바로 파일을 읽을 수 있도록 응용 프로그램 객체를 미리 생성해 놓을 수 있지만 파일의 종류가 많아질수록 미리 생성해야 하는 응용 프로그램 객체 수가 많아지기 때문에 쓸데없이 많은 메모리를 사용하는 프로그램이 된다. 이런 경우 Factory

Method Pattern을 이용하면 사용자가 원하는 시점에서 응용 프로그램 객체를 생성하고 응용 프로그램이 사용자가 선택한 파일 객체를 생성하여 읽을 수 있도록 만들 수 있다.

Factory Method Pattern은 객체를 생성하기 위한 인터페이스를 정의하지만, 어떤 클래스의 인스턴스를 생성할지에 대한 결정은 하위 클래스에서 이루어지도록 인스턴스 생성의 책임을 미룬다. 이 패턴은 기반 클래스가 모든(또는 대부분의) 일을 하지만 정확히 어떤 객체를 갖고 작업할지에 대해서는 런타임 시로 미룰 때 유용하다.

이름	Factory Method Pattern
분류	목적 : 생성 범위 : 클래스
의도	객체를 생성하는 인터페이스를 정의하지만, 인스턴스를 만드는 클래스의 결정은 하위 클래스가 한다. Factory Method Pattern에서는 클래스의 인스턴스를 만드는 시점을 하위 클래스로 미룬다.
구조	 <p>생성해야 하는 객체(Product)와 객체를 생성하는 Creator(Factory)는 인터페이스로 만들어지고 실제 수행을 담당하는 하위 클래스를 갖는 구조이다.</p>
클래스 설명	<ul style="list-style-type: none"> • Creator(Factory): 실제 타입이 알려지지 않은 객체를 생성할 수 있는 메소드로 정의한다. 이를 추상 메소드 호출을 통해 수행한다. • Concrete Creator(ConcreteFactory): Creator의 객체 생성 추상 메소드를 오버라이드하여 Concrete Product를 생성하는 파생 클래스 • Product: 팩토리 메소드가 생성하는 객체의 인터페이스를 정의한다. Creator는 이 인터페이스를 통해 Concrete Product에 접근한다. • Concrete Product: Creator(기반 클래스) 메소드에 의해 사용되는 객체. Product 인터페이스를 구현한다.

예를 들어, 다양한 UI Look And Feel을 만들어 놓고 사용자가 선택하는 시점에서 Look And Feel을 변경할 수 있는 기능을 제공하는 프로그램을 만든다고 가정하자. UI 객체를 생성만을 고려했을 때 프로그램에서 UI 객체가 생성되어야 하는 시점을 알고 있으나 어떤 UI를 생성해야 하는가는 알 수 없는 상황이다. Factory Method Pattern은 객체가 생성되어야 하는 시점은 알고 있으나 어떤 객체를 생성해야 할지 알 수 없을 때 객체의 생성을 하위 클래스에 위임하여 해결한다.

5 Adapter Pattern

소프트웨어 개발을 할 때 관련 라이브러리에서 제공하는 API(Application Program Interface)를 사용하지 않고 개발하는 경우는 드물다. 프로젝트의 기간이 많지 않을 때 관련 라이브러리의 존재는 가뭄의 단비와 같이 개발자에게 기쁜 일이기도 하다. 하지만 사용해야 하는 라이브러리에서 제공하는 API가 내가 개발해서 사용(또는 제공)해야 할 모듈의 인터페이스가 다르다면 난감할 수밖에 없다. 라이브러리 제공자에게 도움을 요청해 바꿀 수 있다면 쉽게 해결되지만 대부분 그럴 가능성은 적다. Adapter Pattern은 이런 경우 적은 비용으로 기존의 라이브러리를 사용할 수 있는 방법을 제시한다.

이름	Adapter Pattern
분류	목적 : 구조 범위 : 클래스, 객체
의도	클래스의 인터페이스를 클라이언트가 기대하는 다른 인터페이스로 변환한다. Adapter Pattern은 호환성이 없는 인터페이스이기 때문에 같이 사용할 수 없는 클래스를 개조하여 함께 작동하도록 해준다.
구조	<p>[상속을 활용한 Adapter Pattern]</p> <p>Adapter 클래스는 Adaptee를 상속 또는 구현(implementation)하여 클라이언트가 사용할 도메인에 종속적인 메소드를 제공하는 구조이다.</p> <p>[객체 합성을 이용한 Adapter Pattern]</p> <p>Adapter 클래스는 Adaptee 클래스 타입의 멤버변수를 선언하여 클라이언트가 사용할 도메인에 종속적인 메소드를 제공한다.</p>

클래스 설명

- Adaptee: Client가 원하는 인터페이스를 지원하지 않는 객체
- Target: Client가 Adaptee로부터 지원을 바라는 오퍼레이션을 가진 인터페이스
- Adapter: Adaptee가 Target 인터페이스를 지원하는 것처럼 보이게 해주는 클래스로 상속이나 위임을 사용하여 구현된다.

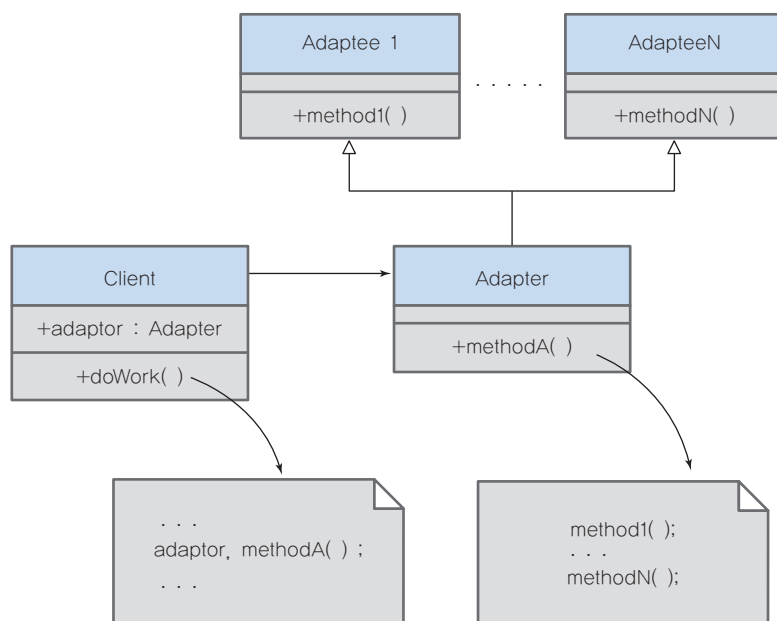
Adapter Pattern이 유용하게 사용되는 경우는 다음과 같다.

- 기존의 클래스를 사용해야 하나 인터페이스가 수정되어야 하는 경우
- 아직 예측하지 못한 클래스나 실제 관련되지 않는 클래스들이 기존의 클래스를 재사용하고자 하지만, 이미 정의된 재사용 가능한 클래스가 지금 요청하는 인터페이스를 꼭 정의하고 있지 않는 경우, 즉 이미 만들어진 것을 재사용하고자 하나 이 재사용 가능한 라이브러리를 수정할 수 없는 경우

Adapter Pattern을 자세히 알아보기 위해 스마트폰 애플리케이션을 개발한다고 가정해 보자.

많은 스마트폰 애플리케이션에 들어가 있는 기능 중 하나는 사진이나 글을 SNS 또는 메신저로 내보내는 기능이다. 사진을 페이스북, 카카오톡, 네이트온 메신저 등으로 공유해서 친구들에게 보여주는 기능이다. 페이스북, 카카오톡, 네이트온 메신저로 사진을 내보내는 기능은 다른 애플리케이션에서도 다시 사용될 수 있는 가능성이 높은 기능이다.

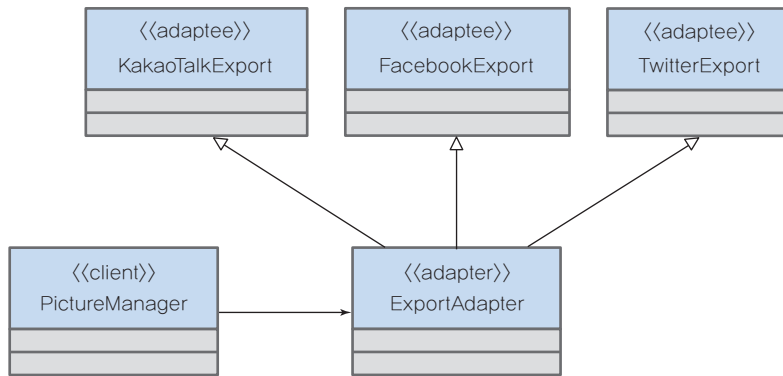
사진 내보내기 기능은 재사용 가능한 기능이긴 하지만, SNS 프로그램에 따라 사진 내보내기 기능의 인터페이스는 모두 다를 수 있다. 이런 경우, 서로 다른 인터페이스의 문제를 해결하기 위한 방법은 무엇이 있을까? 재사용 가능한 모듈에 서로 다른 인터페이스를 갖는 여러 모듈을 결합하고자 할 때 사용할 수 있는 디자인 패턴은 Adapter Pattern이다. 상속을 활용한 Adapter Pattern 구조를 다양한 'SNS'에서 제공하는 API를 수용하는 모습을 나타내는 구조도는 다음과 같다.



::그림 3 | Adapter Pattern의 기본 구조

그림 3에서 Client 클래스는 기존 프로그램을 구성하는 요소이며, Adaptee1에서 AdapteeN까지는 기존 프로그램에 새로 추가되는 클래스이다. Adaptee 클래스들은 동일한 목적의 기능을 가지고 있지만 그 기능을 제공하는 인터페이스는 모두 다르다. 이때 Client 클래스에서 각 Adaptee 클래스의 기능을 호출하고자 한다면, Adaptee 클래스들마다 호출하는 문장을 따로 작성해야 한다. 이러한 문제를 해결해 주기 위해서 Adapter 클래스를 중간에 끼워 넣는 것이 Adapter Pattern의 개념이다. Adapter 클래스는 Adaptee 클래스들의 서로 다른 인터페이스를 하나의 인터페이스처럼 사용할 수 있도록 하는 클래스이다. 그러기 위해서는 Client 클래스는 Adapter 클래스의 메소드를 호출하고, Adapter 클래스의 메소드는 어느 Adaptee 클래스의 메소드를 호출할지 결정한다. Adapter 클래스에서 Adaptee 클래스의 메소드를 호출하는 방법은 어떻게 구현하든 상관없다. 중요한 것은 Adapter 클래스가 서로 다른 Adaptee 클래스들의 인터페이스를 하나처럼 사용할 수 있도록 하는 것이다.

앞에서 설명했던 사례에서 사진 내보내기 기능을 서로 다른 SNS 프로그램에 대해 적용하고자 하는 경우를 클래스 다이어그램으로 나타내면 그림 4와 같다.



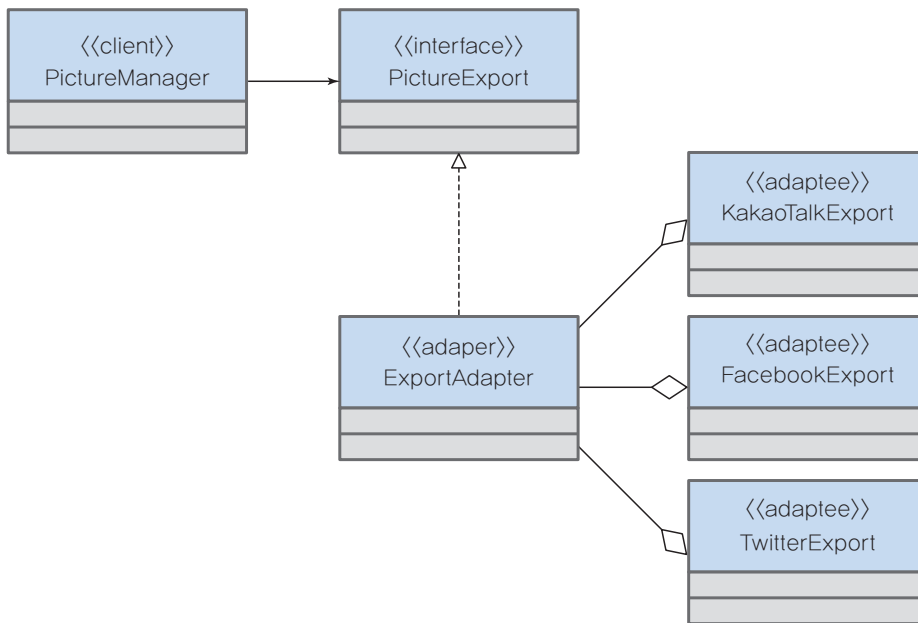
::그림 4 | 사진 내보내기 기능을 위한 Adapter Pattern 구조

그림 4에서 client 클래스인 PictureManager는 여러 SNS 애플리케이션에 사진을 내보내는 역할을 한다. Adaptee 클래스에 해당되는 KakaoTalkExport, FacebookExport, TwitterExport 클래스의 사진 내보내기 메소드를 호출하고자 할 때, 사진 내보내기 메소드의 인터페이스가 모두 다르다면, 그런 인터페이스들을 동일한 형태의 인터페이스로 제공해 줄 수 있는 클래스가 필요하다. 이러한 역할을 수행하는 클래스가 ExportAdapter 클래스이다. PictureManager 클래스는 ExportAdapter 클래스의 사진 내보내기 메소드를 호출할 것이며, 사진 내보내기의 대상이 되는 애플리케이션이 무엇이냐에 따라 적절한 SNS 애플리케이션의 메소드가 호출될 것이다. 이렇게 서로 다른 인터페이스들을 동일한 형태의 인터페이스로 변환해 주는 Pattern이 Adapter Pattern이다.

그림 4와 같은 구조의 Adapter Pattern은 초창기 객체지향 언어의 주를 이루던 C++ 언어에서 사용될 수 있는 클래스 Adapter Pattern이라 볼 수 있다. 그림에서 볼 수 있듯이 Adapter 클래스는 다수의 Adaptee 클래스들로부터 다중 상속을 받고 있다. Adaptee 클래스들의 변형 없이 동일한 형태의 인터페이스를 제공하기 위해서 위와 같은 다중 상속의 구조를 사용하게 되었다. 하지만 다중 상속으로 인한 문제들이 부각되면서 이후의 객체지향 언어인 Java에서는 다중 상속의 개념을 지원하지 않게 되었고, 적어도 Java와 같이 다중 상속을 지원하지 않는 언어에서는 그림 4와 같은 Adapter Pattern을 사용할 수 없게 되었다.

그렇다면 다중 상속의 개념을 지원하지 않는 언어에서는 Adapter Pattern을 사용할 수 없는 것일까? 그렇지 않다. 그림 4와 같이 다중 상속이 지원되지 않는 언어에서는 Adaptee 객체들을 Adapter 클래스가 포함하는 형태로 구성한다. 또한 client 클래스가 여러 Adaptee 클래스들의 메소드를 호출할 수 있도록 하는 target 인터페이스를 두고, Adapter 클래스가 이를 구현하도록 한다. Client 클래스가 target 인터페이스를 통해 사진 내보내기 기능을 호출하면, 해당 메소드를 구현한 Adapter 클래스에서 조건에 따라 적절한 Adaptee 클래스의 메소드

를 호출하는 형태이다. 이러한 구조를 클래스 다이어그램으로 나타내면 그림 5와 같다.



:: 그림 5 | 다중 상속의 개념을 적용하지 않은 Adapter Pattern의 구조

앞에서 언급한 바와 같이 객체지향 언어가 다중 상속의 개념을 지원하느냐 그렇지 않느냐에 따라 Adapter Pattern의 구조가 결정된다고 볼 수 있다.

그림 4와 같이 다중 상속의 개념을 적용한 Adapter Pattern을 ‘클래스 Adapter Pattern’이라 하며, 그림 5와 같이 Adaptee 객체들을 Adapter 클래스가 포함하는 형태의 패턴을 ‘객체 Adapter Pattern’이라 한다.



A p p e n d i x



소프트웨어 개발 산출물 양식

1. 제안 요청서
2. 제안서
3. 소프트웨어 요구사항 명세서
4. 소프트웨어 설계 문서

1 제안 요청서

(1) 제안 요청서(RFP: Request For Proposal)란

명시된 시스템, SW 및 SW 서비스를 발주하기 위하여 입찰 대상자에게 발주자의 요구사항을 알리기 위해 사용하는 문서이며 입찰 공고 시 교부

(2) ‘협상에 의한 계약 체결 기준’ 제2조(기획재정부 회계 예규, 2009. 9. 21)

- ‘제안 요청서’라 함은 계약 담당 공무원이 협상에 의한 계약에 의한 입찰에 참가하고자 하는 자에게 제안서의 제출을 요청하기 위하여 교부하는 서류
- ‘제안서’라 함은 협상에 의한 계약에 의한 입찰에 참가하고자 하는 자가 제안 요청서 또는 입찰 공고에 따라 작성하여 발주자에게 제출하는 서류

(3) 제안 요청서 작성 매뉴얼

- 정보통신산업진흥원(National IT Industry Promotion Agency)은 공공 부문 SW 사업 제안 요청서 작성 매뉴얼을 제공하고 있다.
- 위의 작성 매뉴얼은 정보통신산업진흥원 홈페이지(<http://www.nipa.kr/main.it>)를 통해 다운로드할 수 있으며, 공공 SW사업 제안 요청서 작성 예시가 포함되어 있다.
- 공공 SW사업 제안 요청서는 표 1과 같은 구성으로 작성될 수 있다(예시).

:: 표 1 | 공공 SW 사업 제안 요청서 목차(예시)

사업 개요

- (1) 추진 배경 및 필요성
- (2) 서비스 내용
- (3) 사업 범위
- (4) 기대 효과

현황 및 문제

- (1) 업무 현황
- (2) 정보화 현황
- (3) 문제점 및 개선 방향

사업 추진 방안

- (1) 추진 목표
- (2) 추진 전략
- (3) 추진 체계
- (4) 추진 일정
- (5) 추진 방안

제안 요청 내용

- (1) 제안 요청 개요
- (2) 목표 시스템 개념도
- (3) 개발 대상 업무 내역 및 구성 요건
- (4) 도입 대상 장비 내역 및 구성 요건
- (5) 통합/연계 범위
- (6) 표준 프레임워크 및 공통 컴포넌트 적용
- (7) 초기 자료 구축 요건
- (8) 표준화 요건
- (9) 보안 요건
- (10) 시스템 운용 조건
- (11) 교육 지원 요건
- (12) 기술 지원 요건
- (13) 유지보수 요건
- (14) 기타

사업 개요

(1) 추진 배경 및 필요성

ITA 또는 ISP 결과물, 사업 계획서, 발주 계획서 등을 참조하여 추진 대상 업무의 중요성, 대내외적 환경 여건의 변화, 복잡 다양해지는 사용자 서비스 요구 수준의 고도화 등 사업이 추진되어야 할 배경과 필요성을 제시한다.

(2) 서비스 내용

사업 완료 후 제공될 서비스 및 프로세스(업무 처리 절차) 개선 내용에 대해 서비스 이용 대상자별로 프로세스 개선 전과 개선 후를 명확히 구분하여 향후 제공될 서비스와 그 효과를 기술한다.

(3) 사업 범위

이 사업에서 개발될 응용 SW의 개발 내용 및 범위를 기술하고, 이와 관련된 SW, HW, DB 및 통신망 등 제반 시스템 구축 내용을 기술한다.

(4) 기대 효과

서비스 개발 및 제공으로 발생하는 프로세스 개선 및 비용 절감 효과 등 제반 기대 효과를 제시하고, 가능한 경우 제시된 기대 효과는 정량적 효과와 정성적 효과로 구분하여 기술한다.

현황 및 문제

(1) 업무 현황

- 관련 부처 및 조직, 서비스 대상자 등을 표시하고, 이 사업의 정보 교환 및 상호 간 연계성을 도식화하는 등 이 사업의 전체적인 업무 구성도를 작성한다.
- 사업 범위 내에 포함된 업무명, 업무 개요 및 기능, 수행 조직 및 담당자, 관련 기관, 정보화 여부 등을 상세히 기술한다.
- 업무와 관련된 수행 조직은 조직도를 참조하여 도식화하고 각 조직별 역할을 기술한다.
- 정보 공동 활용 현황을 작성하여 정보 공동 활용이 가능한 자료명 등 정보 공동 활용에 필요한 정보를 기술한다.

(2) 정보화 현황

- 최종 사용자, 통신망, 주전산기, SW, DB로 구성된 전체의 시스템을 도식화하여 현행 시스템 구성도를 작성한다.
- 장비명, 모델, 수량, 용도(주요 기능), 보유 형태, 구입 시기, 설치 장소 등 HW, SW, DB, 통신망 등으로 구분하여 정보화 현황을 기술한다.
- 업무와 관련하여 구축된 DB, 관련 DBMS명 및 논리적 DB 내용, 구축 자료 건수 등을 제공하는 현행 시스템 DB 구축 현황을 작성한다.
- 기타 위에 언급하지 않은 자원 중 이 사업에서 활용될 장비의 품명, 주요 기능, 특징 및 용도 등을 기술한다.

(3) 문제점 및 개선 방향

업무 및 정보화 측면에서의 문제점 및 개선 방안을 기술한다.

사업 추진 방안

(1) 추진 목표

- 사업 추진에 대한 비전, 최종 목표 및 단계별 추진 목표를 제시한다.
- 당해 사업 관련 각종 기본 계획, 시행 계획, 발주 계획서 등을 토대로 작성한다.

(2) 추진 전략

- 이 사업의 목표 달성을 위해 필수적인 전략 및 방향을 제시한다.
- 기술적 측면, 관리적 측면, 정책적 측면, 표준화 측면, 법/제도적 측면 등에서 구체적인 추진 방향 등 선행 사업 수행 전략, 첨단 기술 적용 등 정보 기술 활용 전략, 사업 관리 전략, 표준화 전략, 개발 시스템의 확산 전략, 관련 기관 간 협조, SW 분리 발주 등을 기술한다.

(3) 추진 체계

사업 수행에 필요한 기관(주관 기관, 관련 기관, 사업자 등)을 주관 기관을 중심으로 도식화하고, 담당 업무 및 기능을 상세히 기술한다.

(4) 추진 일정

이 사업의 총 수행 기간, 계약 기간, 추진 체계, 일정별 주요 이벤트(착수 및 종료 보고회, 워크숍, 법 제도 개정, 서비스 개통일, 공청회, 산출물 제출 일정, 장비 및 SW 납품 요구 일정, 설치, 감리) 등을 월별 또는 업무별로 제시한다.

(5) 추진 방안

이 사업 추진의 주요 추진 방안을 제시한다.

제안 요청 내용

(1) 제안 요청 개요

제안 요청 내용 및 주요 핵심 내용을 간략히 요약 기술한다.

(2) 목표 시스템 개념도

목표 시스템을 중심으로 서비스 이용자, 주요 서비스 내역 등을 도식화하여 기술한다.

(3) 개발 대상 업무 내역 및 구성 요건

- 목표 시스템에서 운용될 응용 SW의 개발 범위 및 내용인 업무를 업무 구분, 주요 기능, 세부 설명, 비교 등으로 상세히 표시하고, 응용 SW의 운용에 필요한 구성 요건을 제시한다. 이때, 업무 분석은 기능 점수(FP) 3레벨 수준 이상으로 최대한 자세히 표현해야 한다.
- 주요 제안 요청 내용의 단위 업무별로 과제 개요를 명시하고, 과업 범위별로 주요 기능 및 구축 요건, 서비스 구성도 등을 상세하게 작성한다.

(4) 도입 대상 장비 내역 및 구성 요건

- 목표 시스템 구성을 위해 필요한 도입 대상 HW, SW, 통신망 등의 구성도와 내역(품목, 규격, 수량, 용도, 성능, 처리 용량 등), 필수 요구사항(기능 요건) 및 구성 요건을 제시한다.
- 공개 SW 도입을 저해하는 비표준적인 특정 기술 조건을 명시할 수 없다.
- 분리 발주 대상 SW를 표시한다.
- 제안 장비에 대한 표준 적합성 및 성능 적합성 검증을 실시하고자 하는 경우 각 검증의 대상, 방법, 시기, 절차, 소요 비용 부담 등을 명시한다.

(5) 통합/연계 범위

통합/연계되는 타 기관의 정보 시스템, 기관명, 방법 및 내역을 제시한다.

- 시스템: 목표 시스템과 통합/연계되는 타 기관의 시스템명을 기재한다.
- 기관명: 통합/연계되는 시스템의 대상 기관명을 기재한다.
- 통합/연계 방법: 목표 시스템과 통합/연계되는 방법을 기재한다.
- 통합/연계 내역: 목표 시스템과 통합/연계되는 서비스 및 정보 내용을 기재, 서비스 명 형식으로 기재한다.

(6) 표준 프레임 워크 및 공통 컴포넌트 적용

목표 시스템에 적용할 수 있는 표준 프레임워크 및 공통 컴포넌트와 사용 계획 및 활용 방안을 제시한다.

(7) 초기 자료 구축 요건

목표 시스템의 서비스에 필요한 추가 자료 구축을 위한 목적, 대상, 역할 분담, 구축 내역 및 구축 방안과 절차 등을 제시한다.

(8) 표준화 요건

표준화 항목에 맞는 사업 내역을 제시한다.

(9) 보안 요건

해당 사업 수행과 관련하여 생성된 문서의 보관, 통신 보안, 시스템 보안 등의 보안 대책과 개인 정보 보호 대책을 제시한다.

(10) 시스템 운용 조건

목표 시스템의 정상 운용을 위한 시스템적인 조건, 조직, 보안 대책 등을 기술한다.

(11) 교육 지원 요건

목표 시스템의 사용자, 관리자, 운용자 등 시스템의 이용 대상자별 교육 내용, 교육 기간, 인원, 횟수, 교육장의 지방 소재 여부, 장소 제공 여부, 강사 조달 주체, 비용 부담 여부, 온라인 교육 대체 가능 여부, 사용자 매뉴얼 여부 등을 제시한다.

(12) 기술 지원 요건

목표 시스템과 관련하여 기술 지원 대상 범위, 기술 지원 내용 및 수준, 기술 매뉴얼, 헬프 데스크(help desk) 등 기술 지원이 필요한 사항에 대한 내용을 기술한다.

(13) 유지보수 요건

목표 시스템의 HW, SW의 유지보수 활동을 위한 업무 처리 등 관련 사항은 기획재정부 회계 예규 용역 계약 일반 조건 제58조를 준용하여 제시한다.

(14) 기타

- 위에 제시되지 않은 기타 요구사항을 기술한다.
- 필수 기재 사항: SW 사업 수행을 위한 장소 및 설비 기타 작업 환경에 대한 주관 기관 제공/사업자 부담 여부를 명시한다.
- 사업 범위 내 업무 현황/문제점/이슈/개선 방안과 관련하여 사업 계획서 등에서 분석된 자료를 기술하거나 첨부로 제시할 수 있다.

2 제안서

(1) 제안서란

제안서(Proposal)는 서비스(또는 제품)를 개발할 기업이 발주자가 의뢰한 프로젝트를 어떻게 수행할 것인지를 제안 요청서(RFP)를 근거로 정리한 문서이며, 입찰 시 발주자에게 제출한다.

(2) ‘협상에 의한 계약 체결 기준’ 제2조(기획재정부 회계 예규, 2009. 9. 21)

- ‘제안 요청서’라 함은 계약 담당 공무원이 협상에 의한 계약에 의한 입찰에 참가하고자 하는 자에게 제안서의 제출을 요청하기 위하여 교부하는 서류
- ‘제안서’라 함은 협상에 의한 계약에 의한 입찰에 참가하고자 하는 자가 제안 요청서 또는 입찰 공고에 따라 작성하여 계약 담당 공무원에게 제출하는 서류

(3) 제안서 작성 매뉴얼

- 제안 요청에서는 제안서 작성 방법을 제시하므로 각각의 제안 요청서의 성격에 따라 제안서의 작성 방법이 다를 수 있다.

- 다음의 제안서 목차 및 설명은 정보통신산업진흥원에서 공고한 ‘e-Cube센터 정보자원, 보안 및 PC유지관리 용역’ 제안 요청서의 제안 안내서를 참고하였다.
- 제안서는 표 2와 같은 구성으로 작성될 수 있다(예시).

:: 표 2 | 제안서 작성 목차(예시)

제안 개요	프로젝트 관리
일반 현황 및 유사 사업 경험	(1) 관리방법론
수행 조직 및 인원	(2) 일정 계획
(1) 수행 조직 및 업무 분장	(3) 운영 환경
(2) 투입 인력 및 이력사항	프로젝트 지원
전략 및 방법론	(1) 유지 관리 계획
(1) 사업 수행 전략	(2) 교육 훈련
(2) 적용 기술	(3) 기술 이전 계획
(3) 운영방법론	(4) 기타 지원 사항
기술 부문	가격 제안서(별첨)
(1) 용역 수행 방안	기타
(2) 유지 관리 수행 조건	
(3) 인력 요구사항	
(4) 보안 요구사항	
(5) 품질 요구사항	
(6) 계약 충족 방안	
(7) 프로젝트 관리 요구사항	
(8) 프로젝트 지원 요구사항	
(9) 향후 시스템 운영 발전 방향	

제안 개요

제안 요청서 내용을 명확히 이해하고 제안의 목표, 범위, 전제 조건, 제안의 특징을 요약하여 기술한다.

일반 현황 및 유사 사업 경험

제안사의 일반 현황 및 주요 연혁, 최근 3년간의 자본금 및 부문별 매출액, 주요 사업 내용, 주요 사업 실적, 조직 및 인원 현황을 기술한다. 해당 사업과 관련이 있는 최근 3년간의 주요 사업 실적(건수, 기간, 규모, 역할 등)을 기술한다.

수행 조직 및 인원

(1) 수행 조직 및 업무 분장

이 사업을 수행할 조직 및 업무분장 내용을 상세히 기술한다.

(2) 투입 인력 및 이력 사항

이 사업을 수행할 인력을 작업 단위별로 제시하고, 투입 인력에 대한 이력 사항을 붙임 양식을 이용하여 작성한다.

전략 및 방법론

(1) 사업 수행 전략

사업을 효과적으로 수행하기 위한 추진 전략, 적용 기술 등 계획을 기술한다.

(2) 적용 기술

사업에서 적용하고자 하는 기술의 향후 확장성 등을 기술한다.

(3) 운영 방법론

업무에 적용할 방법론의 활용 방안을 제시하며, 적용 방법론의 경험을 기술한다.
제출할 산출물의 종류 및 내역, 제출 시기를 기술한다.

기술 부문(기술 및 기능, 성능 및 품질 부문 등 요구사항에 대한 충족 여부)

(1) 용역 수행 방안

이 사업 수행을 위한 추진 체계, 추진 전략, 추진 내용 등을 기술하고 시스템 운영 장소가 다수일 경우 운영 장소별 추진 내용을 제시한다.

(2) 유지 관리 수행 조건

유지보수 일반 사항, 기술 사항, 유지보수 수행 조건, 장애 관리, 훈련 실시, 마스터데이터 운영 및 장애 관리, WAS 운영 및 장애 관리 내용을 제시한다.

(3) 인력 요구사항

시스템 운영을 위한 상주 및 비상주 인력 운영 계획을 제시한다.

(4) 보안 요구사항

계약사, 인원 보안 사항 및 시스템 등의 보안 요구사항 준수 대책을 제시하며 홈페이지 보안

취약점 점검, 보안 장비 운영 및 관제, 마스터데이터 DB 보안 준수 등의 방안을 제시한다.

(5) 품질 요구사항

유지보수 시간, 장애 복구, 분기별 데이터 품질 분석 등의 방안을 제시한다.

(6) 제약 충족 방안

부품 지원, 이전 지원, 손해배상 책임 등에 관한 방안을 제시한다.

(7) 프로젝트 관리 요구사항

프로젝트 관리 일반 사항, 비상 연락망 관리, 안전 관리 등의 방안을 제시한다.

(8) 프로젝트 지원 요구사항

교육 지원, 기술 지원, 인수 인계, 자원 현황, 표준 준수 등의 방안에 대하여 기술한다.

(9) 향후 시스템 운영 발전 방향

이 시스템 운영과 관련 향후 발전 방향에 대하여 기술한다.

프로젝트 관리

(1) 관리방법론

위험 관리 방안, 자원 관리 방안, 진도 관리 방안, 커뮤니케이션 방안, 보안 관리 방안, 형상 관리 방안, 문서 관리 방안, 변화 관리 방안을 기술한다.

(2) 일정 계획

사업 추진 예정 일정을 참조하여 추진 일정을 상세히 기술한다.

추진 일정에는 단계별 목표를 정의하고 자원 배분 계획을 포함한다.

(3) 운영 환경

운영 장비 및 도구의 보유 현황 및 확보 방안, 운영 공간 및 인력에 대한 지원 및 관리 방안 등을 기술한다.

프로젝트 지원

(1) 유지 관리 계획

유지보수 및 기능 향상, 장애 처리를 위한 종합적인 방안을 제시한다.

(2) 교육 훈련

시스템 운영자의 기술 향상을 위하여 교육 훈련 추진 방향 및 유·무상 교육 지원 계획을 제시한다.

(3) 기술 이전 계획

원활한 시스템 운영을 위해 유지보수 요원에 대한 기술 이전 계획을 분야별로 상세하게 제시한다.

(4) 기타 지원 사항

이 사업과 관련 지원 가능한 사항에 대한 내용을 기술한다.

가격 제안서(별첨)

전체 사업에 필요한 부분별 가격을 제시한다.

기타

위 항목에서 제시되지 않은 기타 내용을 기술한다.

참고 자료: 정보통신산업진흥원(www.nipa.kr) >입찰 공고 >공고 번호 : NIPA 제13-412호
'e-Cube센터 정보자원, 보안 및 PC유지관리 용역' 제안 안내서

3 소프트웨어 요구사항 명세서

(1) 소프트웨어 요구사항 명세서의 정의

소프트웨어 요구사항 명세서(SRS: Software Requirements Specification): 고객의 요구사항(시스템이 수행하는 기능)을 정의하여 명시해 놓은 문서

(2) 소프트웨어 요구사항 명세서 작성 매뉴얼

- IEEE(IEEE std. 830-1998 Recommended Practice for Software Requirements Specifications)는 소프트웨어 요구사항 명세서 작성 매뉴얼을 제공하고 있다.
- 한국정보통신기술협회(Telecommunications Technology Association)는 소프트웨어 요구사항 명세서 표준(TTAS_KO-11_0022)을 제공하고 있다. 이 표준은 IEEE Std 830-1998 표준을 참조하여 작성되었다.
- 위의 작성 매뉴얼은 각각 한국정보통신기술협회 및 IEEE 홈페이지를 통해 다운로드할 수 있다.
- 소프트웨어 요구사항 명세서는 표 3과 같은 구성으로 작성할 수 있다(예시).

:: 표 3 | 소프트웨어 요구사항 명세서 목차(예시)

서론	세부 요구사항
(1) 목적	(1) 외부 인터페이스
(2) 범위	(2) 기능
(3) 정의, 머리글자어, 약어	(3) 성능 요구사항
(4) 참고 문헌	(4) 논리적 데이터베이스 요구사항
(5) 개요	(5) 설계상의 제약 사항
총괄 기술	(6) 소프트웨어 시스템 속성
(1) 제품의 조망	(7) 세부 요구사항의 구성
(2) 제품의 기능	지원 정보
(3) 사용자 특성	(1) 목차 및 색인
(4) 제약 사항	(2) 부록
(5) 가정과 의존사항	
(6) 요구사항의 유보	

서론

(1) 목적

소프트웨어 요구사항 명세서의 목적을 서술하고, 대상 독자를 명시한다.

(2) 범위

생산될 소프트웨어의 제품을 이름으로 식별한다. 소프트웨어 제품이 할 일, 필요하다면 하지 말아야 할 일을 설명한다. 장점, 목적, 목표 등을 포함하여 소프트웨어의 애플리케이션을 기술하되, 상위 수준의 명세서가 존재한다면 모순된 점이 없도록 상위 명세서의 내용과 일치하여야 한다.

(3) 정의, 머리글자어, 약어

소프트웨어 요구사항 명세서를 정확하게 해석하는 데 필요한 모든 용어들을 정의하고, 머리글자어와 약어들을 제공한다.

(4) 참고 문헌

소프트웨어 요구사항 명세서 내에서 참조되는 모든 문서들의 완전한 목록을 제시한다. 문서의 제목, 보고서 번호, 날짜, 출판사 등을 이용하여 각 문서들을 식별하고 참고 문서들을 얻을 수 있는 출처를 기술한다.

(5) 개요

소프트웨어 요구사항 명세서의 나머지 부분의 내용을 기술하며, 명세서가 어떻게 구성되어 있는지를 설명한다.

총괄 기술

(1) 제품의 조망

① 시스템 인터페이스

각각의 시스템 인터페이스를 열거하고, 시스템 요구사항을 달성하기 위한 소프트웨어 기능을 파악하며 시스템을 연결하기 위한 인터페이스를 찾아내어 기술한다.

② 사용자 인터페이스

소프트웨어 제품과 사용자 간의 각 인터페이스들의 논리적 특성을 기술한다. 소프트웨어 요구를 달성하기 위해 필요한 형상 특성들이 포함된다(예를 들면, 화면 형식, 쪽이나 윈도우 배치 등).

③ 하드웨어 인터페이스

소프트웨어 제품과 시스템의 하드웨어 구성 요소 간의 인터페이스에 대한 논리적 특성을 기술한다.

④ 소프트웨어 인터페이스

필요한 소프트웨어 제품들의 사용(예를 들면, 데이터베이스 관리 시스템, 운영체제, 수학 패키지 등)과 응용 시스템과의 인터페이스를 기술한다. 요구되는 소프트웨어 제품의 정보(이름, 기억 코드, 규격 번호, 버전 번호, 출처 등)가 제공되어야 한다.

⑤ 통신 인터페이스

지역 네트워크 프로토콜과 같은 다양한 통신 인터페이스를 기술한다.

⑥ 기억 장치

주 기억 장치와 보조 기억 장치에 대한 적용 특성과 제한 사항들을 기술한다.

⑦ 운영

정상적인 경우나 특별한 경우의 운영 사항을 기술한다.

⑧ 사이트 적용 요구사항

주어진 사이트에 맞는 데이터와 초기화 절차 및 임무, 운영 모드들에 대한 요구사항을 정의한다. 특정 설치 요구에 소프트웨어를 적용하기 위해 수정해야 하는 사이트 또는 임무 관련 사항들을 기술한다.

(2) 제품의 기능

이 절은 소프트웨어가 수행할 주요 기능들의 요약を提供한다. 때때로 이 부분에 필요한 기능 요약은 상위 수준의 명세서로부터 직접 가져오기도 한다. 제품의 기능을 설명할 때에는 고객이나 이 문서를 처음 읽는 다른 사람들이 쉽게 기능을 이해할 수 있도록 목록을 만들고, 제품의 설계 내용을 보여주는 것이 아닌 변수들 간의 논리적 관계만을 보여주도록 한다.

(3) 사용자 특성

이 절은 교육 수준, 경험, 기술적 숙련도 등을 포함하여 제품 사용자들의 일반적인 특성을 기술한다.

(4) 제약 사항

이 절은 개발자의 선택을 제한하는 다른 항목들을 기술한다. 예를 들면, 규정 정책, 하드웨어 제한 사항, 안전 및 보안 고려 사항 등이 있다.

(5) 가정과 의존 사항

소프트웨어 요구사항 명세서에 기술된 요구사항에 영향을 미치는 요인들을 나열한다. 이 요인들은 소프트웨어의 설계에 대한 제약 사항이 아니고, 요구사항에 영향을 미칠 수 있는 어떤 요인들에 대한 변경 사항들이다.

(6) 요구사항의 유보

이 절은 시스템의 다음 버전이 나올 때까지 개발이 연기되는 요구사항들을 기술한다.

세부 요구사항

이 절에서는 설계자들이 이 요구사항을 만족하는 시스템을 설계하는 데 충분하도록, 그리고 테스터들이 이 시스템이 주어진 요구사항을 제대로 만족하는지를 시험해 볼 수 있도록 충분히 상세한 수준의 소프트웨어 요구사항을 서술한다. 사용자와 운영자 또는 다른 외부의 시스템들은 이 절을 통해서 작성된 모든 요구사항들을 확실히 이해할 수 있어야 한다.

이 절에 서술되는 요구사항들은 최소한 시스템에 들어오는 모든 입력과 출력, 시스템에 의해 수행되는 모든 기능들을 포함해야 한다. 이 부분은 소프트웨어 요구사항 명세서에서 가장 비중이 크고 중요한 부분이기 때문에 다음의 원칙을 준수하도록 한다. 첫째, 세부 요구사항들은 전에 작성된 관련 문서와 상호 참조되어야 한다. 둘째, 모든 요구사항들은 유일하게 식별되어야 한다. 셋째, 판독성이 최대화되도록 요구사항들을 구성하는 데 세심한 주의를 기울여야 한다.

(1) 외부 인터페이스

소프트웨어 시스템에 들어오고 나가는 모든 입력과 출력을 상세히 기술한다.

(2) 기능

기능 요구사항은 입력을 받아서 처리하여 출력을 생성하는 데 있어 소프트웨어에서 발생하는 기본적인 행동을 정의한다.

(3) 성능 요구사항

소프트웨어 또는 소프트웨어와 사람 간의 상호작용상 존재하는 정적·동적 요구사항을 모두 계량적으로 기술한다.

(4) 논리적 데이터베이스 요구사항

데이터베이스에 저장될 정보의 논리적 요구사항에 대해 기술한다.

(5) 설계상의 제약 사항

다른 표준이나 하드웨어의 제한 때문에 생길 수 있는 설계상의 제약사항들을 기술한다.

(6) 소프트웨어 시스템 속성

신뢰성, 가용성, 보안성, 유지보수성, 이식성 등 요구사항으로 제시해야 할 소프트웨어 속성들을 기술한다. 요구된 속성들이 나중에 달성되었는지를 검증할 수 있도록 하는 것이 중요하다.

(7) 세부 요구사항의 구성

일상적인 평범한 시스템을 제외하면 대부분의 시스템은 세부 요구사항이 계속 확장되는 경향이 있다. 이러한 이유로, 이해하기 가장 좋은 방법으로 요구사항을 구성하기 위하여 주의 깊게 고려하도록 권고하고 있다. 모든 시스템에 맞는 최적의 구조는 없기 때문에 여기서는 그 구조의 종류만 언급하도록 한다. 해당 구조의 예시나 자세한 내용을 확인하고자 할 경우에는 소프트웨어 요구 명세서 표준(TTAS_KO-11_0022)의 부록을 참고한다.

- 시스템 모드
- 사용자 범주
- 객체
- 특징
- 자극
- 반응
- 기능 계층 구조

지원 정보

(1) 목차 및 색인

일반적인 작성 원칙에 따라 작성한다.

(2) 부록

부록은 항상 실제 요구사항 명세서의 일부분으로 간주되는 것은 아니며, 늘 필요한 것도 아니다. 다음과 같은 사항을 포함할 수 있다.

- 견본 입/출력 형식, 비용 분석 연구에 대한 설명 또는 사용자 조사 결과
- 소프트웨어 요구사항 명세서를 읽는 사람들을 도와줄 수 있는 지원 정보나 배경
- 소프트웨어에 의해 해결되어야 하는 문제의 설명
- 보안, 배포, 초기 설치 또는 다른 요구사항들을 만족시키기 위한 코드와 매체에 대한 특별한 묶음 명령어들

부록이 포함될 때는 부록이 요구사항의 일부분인지 또는 아닌지에 대해 명확히 언급하도록 한다.

참고 문헌: 한국정보통신기술협회, “소프트웨어 요구명세서 표준”(TTAS_KO-11_0022)

4 소프트웨어 설계 문서

소프트웨어 설계 문서(SDD: Software Design Document)는 생성할 소프트웨어 시스템의 모형 또는 표현이다. 이 모형은 소프트웨어 시스템의 계획, 분석, 구현에 필요한 정확한 설계 정보를 제공해야 한다. 이는 시스템이 분할된 설계 엔티티들로 표현되고 이들의 중요한 속성과 이들 엔티티 사이의 관계를 기술해야 한다. 하나의 소프트웨어 시스템을 표현하는 데 사용되는 설계 기술서 모형은 설계 엔티티의 집합, 각 소유 속성들과 관계들로 표현될 수 있다.

- 한국정보통신기술협회(Telecommunications Technology Association)는 소프트웨어 설계 문서 표준(TTAS_KO-11_0023)을 제공하고 있다. 이 표준은 IEEE Std 1016-1998 표준을 참조하여 작성되었다.
- 위의 작성 매뉴얼은 한국정보통신기술협회 홈페이지(<http://www.tta.or.kr/index.jsp>)를 통해 다운로드할 수 있다.
- 소프트웨어 설계 문서는 표 4와 같은 구성으로 작성될 수 있다(예시).

::표 4 | 소프트웨어 설계 문서 목차(예시)

개요	의존성 기술
(1) 목적	(1) 내부 모듈 의존성
(2) 범위	(2) 내부 프로세스 의존성
(3) 정의 및 두문어	(3) 데이터 의존성
참고 문헌	인터페이스 기술
분해 기술	(1) 모듈 인터페이스
(1) 모듈 분해	① 모듈 1 설명
① 모듈 1 설명	② 모듈 2 설명
② 모듈 2 설명	(2) 프로세스 인터페이스
(2) 병행 프로세스 분해	① 프로세스 1 설명
① 프로세스 1 설명	② 프로세스 2 설명
② 프로세스 2 설명	상세 설계
(3) 데이터 분해	(1) 모듈 상세 설계
① 데이터 엔티티 1 설명	① 모듈 1 상세 사항
② 데이터 엔티티 2 설명	② 모듈 2 상세 사항
	(2) 데이터 상세 설계
	① 데이터 엔티티 1 상세 사항
	② 데이터 엔티티 2 상세 사항

개요

설계 문서가 작성된 목적과 적용 범위 그리고 설계 문서에서 사용된 용어에 대한 정의와 두문어에 대한 설명이 포함된다.

참고 문헌

설계 문서 작성에 있어서 참조된 문서들의 목록을 기술한다. 이전 단계에서 작성된 문서 (요구사항 명세서, 설계 표준 문서 등)들이 포함된다.

분해 기술

소프트웨어 분해 설명은 소프트웨어 시스템을 설계 요소로 나누어 기록한다. 이는 시스템이 구조화된 방식과 각 요소의 목적과 기능을 포함한다. 분해 설명은 시스템의 주요 설계 항목을 식별하기 위해서 요소가 특정한 기능들을 이행할 책임이 있는지 결정하고 설계 요소에 대하여 요구사항을 추적할 목적으로 사용될 수 있다. 분해 기술 정보는 소프트웨어 프로젝트 계획, 감시, 제어를 위해 사용될 수 있다. 이 설계 정보는 각각의 소프트웨어 구성 요소, 목적, 기본 기능을 식별하고, 다른 프로젝트 정보와 함께 비용과 인원에 대한 산정, 개발 노력에 대한 일정을 예측하는 데 사용될 수 있다. 시스템 분해를 설명하기 위해 사용되는 주요 그래픽 기법은 계층적 분해 다이어그램이다. 이 다이어그램은 각 요소에 대한 목적과 기능을 기술하는 자연어와 함께 사용될 수 있다.

의존성 기술

의존성 기술은 요소 간의 관계를 기술한다. 이것은 의존 관계에 있는 요소들을 인식하고 이들의 결합도를 기술하며 요구되는 자원들을 정의한다. 의존성 기술은 요구사항과 설계 사항이 변경될 때, 그 영향을 평가하기 위하여 시스템이 어떻게 작업하는지에 대한 전반적인 그림을 제공한다. 이것은 시스템 고장 또는 자원 병목현상을 일으키는 요소들을 분리하도록 도와줄 수 있다. 이 기술서는 통합 시험 케이스를 생성하는 데 도움을 주기 위해서 통합 시험에서 사용될 수 있다. 설계 요소들 간의 관계는 자료 흐름도, 구조도 또는 트랜잭션 다이어그램으로 표현된다.

인터페이스 기술

요소 간의 인터페이스 기술은 각 요소에 의해 제공되는 기능을 올바르게 사용되는지 알고 하는 설계자, 프로그래머, 시험자에게 필요한 것들을 제공한다. 이것은 소프트웨어 요구 사항 명세서에서 제공하지 않은 외적 · 내적 인터페이스의 자세한 내용을 포함한다. 이 장에서는 각 요소에 대한 인터페이스 명세의 집합으로 이루어진다. 인터페이스 기술은 설계자, 프로그래머, 고객 및 테스터 간의 구속력 있는 계약 사항으로 작용한다. 인터페이스 기술은 화면 포맷, 유효한 입력, 결과 출력을 가지는 각각의 요소와 통신할 수 있는 언어를 제공해야 한다.

상세 설계

상세 설계 기술은 각각의 설계 요소에 대한 내부 상세 사항을 포함한다. 이들 상세 사항은 정의, 처리, 데이터에 대한 속성 기술을 포함한다. 이 속성 정보는 모든 설계 요소들을 위해 제공되어야 한다. 이 설계 기술은 구현 전에 프로그래머에게 필요한 상세 사항을 포함한다. 상세 설계 기술은 단위 시험 계획을 생성하는 데 도움을 주기 위해 사용될 수도 있다. 설계 요소의 상세 사항을 기술하기 위하여 사용되는 많은 도구들이 있다. 프로그램 설계 언어는 요소를 위한 입력, 출력, 지역 데이터, 알고리즘을 기술하기 위해 사용될 수 있다. 설계 요소 논리를 기술하기 위한 사용 기법은 메타 코드, 구조적 언어 또는 순서도 같은 그래픽 방법들이 있다.