

대분류/20  
정보통신

중분류/01  
정보기술

소분류/02  
정보기술개발

세분류/02  
응용SW엔지니어링

능력단위/26

NCS학습모듈

# 애플리케이션 테스트

## 관리

LM2001020226\_16v4



교육부

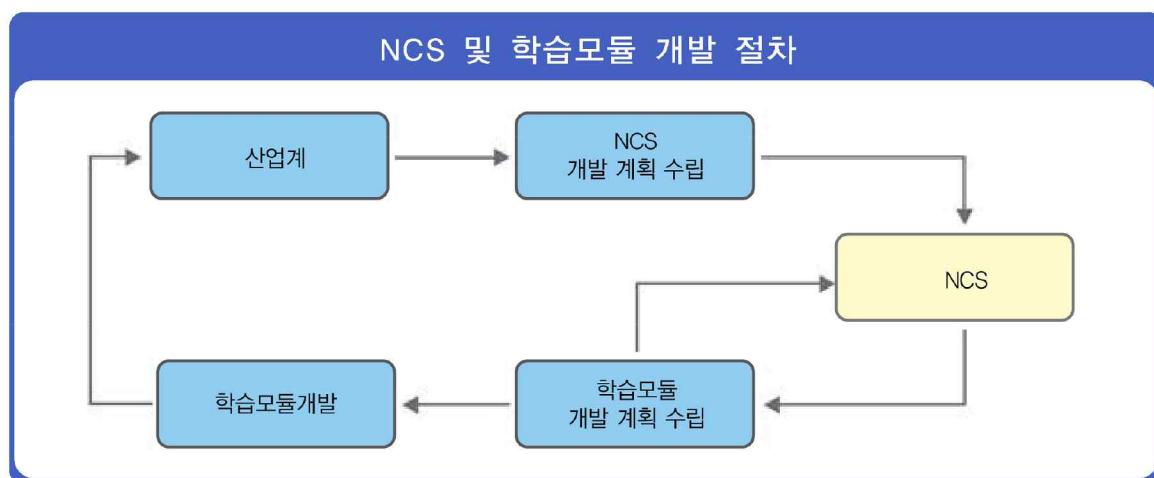
NCS 학습모듈은 교육훈련기관에서 출처를 명시하고 교육적 목적으로 활용할 수 있습니다. 다만 NCS 학습모듈에는 국가(교육부)가 저작재산권 일체를 보유하지 않은 저작물들(출처가 표기되어 있는 도표, 사진, 삽화, 도면 등)이 포함되어 있으므로 이러한 저작물들의 변형, 복제, 공연, 배포, 공중 송신 등과 이러한 저작물들을 활용한 2차 저작물의 생성을 위해서는 반드시 원작자의 동의를 받아야 합니다.

# NCS 학습모듈의 이해

\* 본 학습모듈은 「NCS 국가직무능력표준」 사이트(<http://www.ncs.go.kr>)에서 확인 및 다운로드 할 수 있습니다.

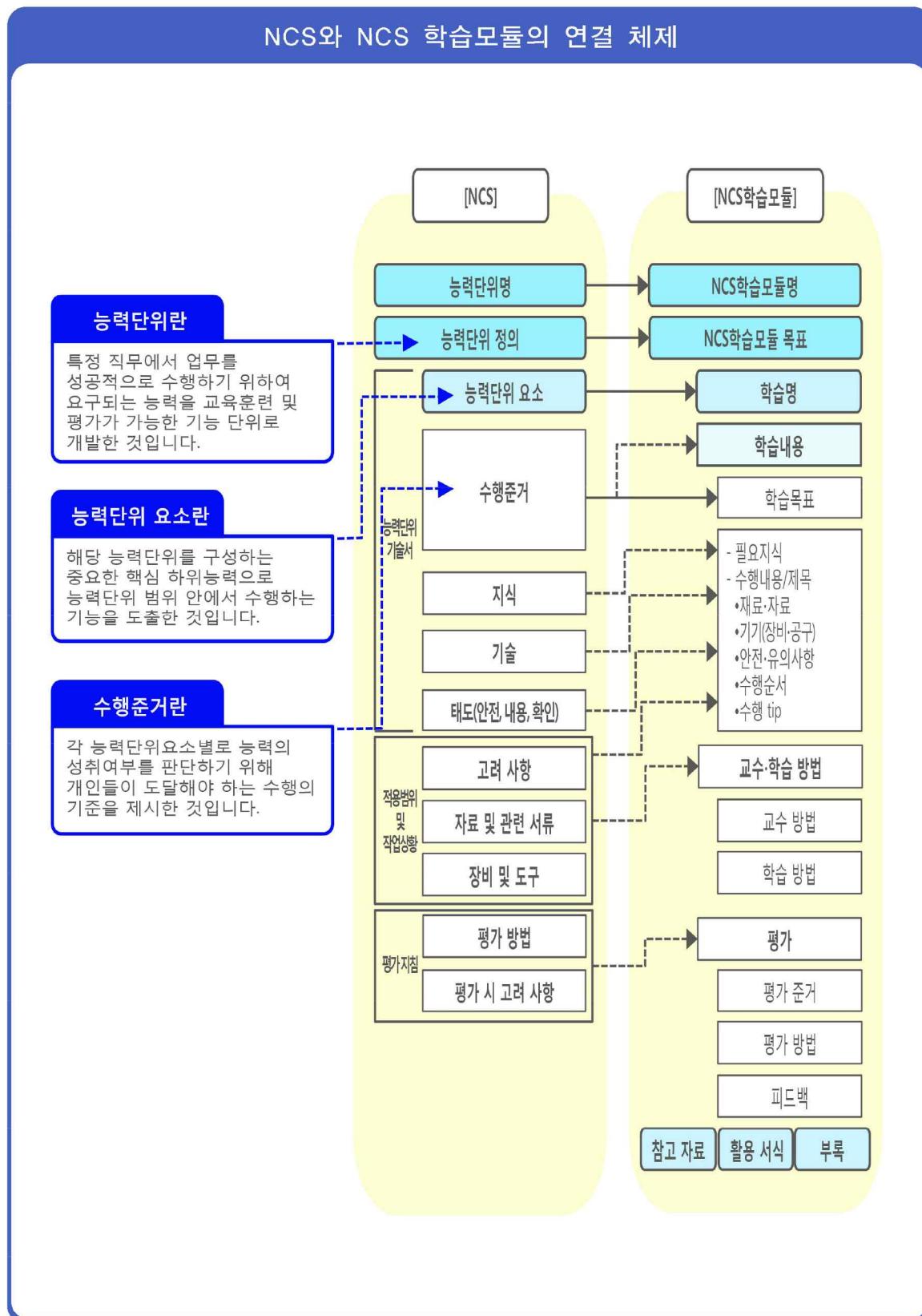
## (1) NCS 학습모듈이란?

- 국가직무능력표준(NCS: National Competency Standards)이란 산업현장에서 직무를 수행하기 위해 요구되는 지식·기술·소양 등의 내용을 국가가 산업부문별·수준별로 체계화한 것으로 산업현장의 직무를 성공적으로 수행하기 위해 필요한 능력(지식, 기술, 태도)을 국가적 차원에서 표준화한 것을 의미합니다.
- 국가직무능력표준(이하 NCS)이 현장의 ‘직무 요구서’라고 한다면, NCS 학습모듈은 NCS의 능력단위를 교육훈련에서 학습할 수 있도록 구성한 ‘교수·학습 자료’입니다. NCS 학습모듈은 구체적 직무를 학습할 수 있도록 이론 및 실습과 관련된 내용을 상세하게 제시하고 있습니다.



- NCS 학습모듈은 다음과 같은 특징을 가지고 있습니다.
  - 첫째, NCS 학습모듈은 산업계에서 요구하는 직무능력을 교육훈련 현장에 활용할 수 있도록 성취목표와 학습의 방향을 명확히 제시하는 가이드라인의 역할을 합니다.
  - 둘째, NCS 학습모듈은 특성화고, 마이스터고, 전문대학, 4년제 대학교의 교육기관 및 훈련기관, 직장교육기관 등에서 표준교재로 활용할 수 있으며 교육과정 개편 시에도 유용하게 참고할 수 있습니다.

- NCS와 NCS 학습모듈 간의 연결 체계를 살펴보면 아래 그림과 같습니다.



## (2) NCS 학습모듈의 체계

- NCS 학습모듈은 1.학습모듈의 위치, 2.학습모듈의 개요, 3.학습모듈의 내용 체계, 4.참고 자료, 5.활용 서식/부록으로 구성되어 있습니다.

### 1. NCS 학습모듈의 위치

- NCS 학습모듈의 위치는 NCS 분류 체계에서 해당 학습모듈이 어디에 위치하는지를 한 눈에 볼 수 있도록 그림으로 제시한 것입니다.

예시 : 이·미용 서비스 분야 중 네일미용 세분류

### NCS-학습모듈의 위치

대분류	이용·숙박·여행·오락·스포츠
중분류	이·미용
소분류	아·미용 서비스

#### 세분류

헤어미용
피부미용
메이크업
<b>네일미용</b>

#### 능력단위('17.4. 고시)

네일 샵 위생 서비스	네일숍 위생서비스
네일 화장물 제거	네일 화장물 제거
<b>네일 기본 관리</b>	<b>네일 기본관리</b>
네일 렌	네일 랩
네일 팁	네일 팁
젠퐁 네일	젠퐁 네일
아크릴릭 네일	아크릴 네일
평면 네일아트	평면 네일아트
융합 네일아트	융합 네일아트
네일 샵 운영관리	네일숍 운영관리

#### 학습모듈명

#### 학습모듈은

NCS 능력단위 1개당 1개의 학습모듈 개발을 원칙으로 합니다. 그러나 필요에 따라 고용 단위 및 교과 단위를 고려하여 능력단위 몇 개를 묶어서 1개의 학습모듈로 개발할 수 있으며, NCS 능력단위 1개를 여러 개의 학습모듈로 나누어 개발할 수도 있습니다. 또한 보완 개발 세분류의 경우 일부 보완된 능력단위에 한해 학습모듈을 보완 개발하는 경우가 있으며, 이는 학습모듈명 우측에 있는 \*표기로 확인할 수 있습니다.

## 2. NCS 학습모듈의 개요



### 구성

- NCS 학습모듈 개요는 학습모듈이 포함하고 있는 내용을 개략적으로 설명한 것으로서 **학습모듈의 목표**, **선수 학습**, **학습모듈의 내용 체계**, **핵심 용어**로 구성되어 있습니다.

#### 학습모듈의 목표

해당 NCS 능력단위의 정의를 토대로 학습목표를 작성한 것입니다.

#### 선수 학습

해당 학습모듈에 대한 효과적인 교수·학습을 위하여 사전에 이수해야 하는 학습모듈, 학습 내용, 관련 교과목 등을 기술한 것입니다.

#### 학습모듈의 내용 체계

해당 NCS 능력단위요소가 학습모듈에서 구조화된 방식을 제시한 것입니다.

#### 핵심 용어

해당 학습모듈의 학습 내용, 수행 내용, 설비·기자재 등 가운데 핵심적인 용어를 제시한 것입니다.



### 활용 안내

예시 : 네일미용 세분류의 ‘네일 기본관리’ 학습모듈

#### 네일 기본관리 학습모듈의 개요

##### 학습모듈의 목표

고객의 네일 보호와 미적 요구 충족을 위하여 효과적인 네일 관리로 프리에지 형태 만들기, 큐티를 정리하기, 컬러링하기, 보습제 도포하기, 마무리를 할 수 있다.

##### 선수학습

네일숍 위생서비스(LM1201010401\_14V2)

##### 학습모듈의 내용체계

학습	학습 내용	NCS 능력단위 요소	
		코드번호	요소 명칭
1. 프리에지 형태 만들기	1-1. 네일 파일에 대한 이해와 활용 1-2. 프리에지 형태 파일링	1201010403_12v2.1	프리에지 모양 만들기
2. 큐티를 정리하기	2-1. 네일 기본관리 매뉴얼 이해 2-2. 큐티를 관리	1201010403_14v2.2	큐티를 정리하기
3. 컬러링하기	3-1. 컬러링 매뉴얼 이해 3-2. 컬러링 방법 선정과 작업	1201010403_14v2.3	컬러링
4. 보습제 도포하기	4-1. 보습제 선정과 도포 4-2. 각질제거	1201010403_14v2.4	보습제 바르기
5. 네일 기본관리 마무리하기	5-1. 유분기 제거 5-2. 네일 기본관리 마무리와 청리	1201010403_14v2.5	마무리하기

##### 학습모듈의 목표는

학습자가 해당 학습모듈을 통해 성취해야 할 목표를 제시한 것으로, 교수자는 학습자가 학습모듈의 전체적인 내용흐름을 파악할 수 있도록 지도하는 것이 필요합니다.

##### 선수 학습은

교수자나 학습자가 해당 모듈을 교수 또는 학습하기 이전에 이수해야 할 학습내용, 교과목, 핵심 단어 등을 표기한 것입니다. 따라서 교수자는 학습자가 개별 학습, 자기 주도 학습, 방과 후 활동 등 다양한 방법을 통해 이수할 수 있도록 지도하는 것이 필요합니다.

##### 핵심 용어는

학습모듈을 통해 학습되고 평가되어야 할 주요 용어입니다. 또한 당해 모듈 또는 타 모듈에서도 핵심 용어를 사용하여 학습내용을 구성할 수 있으며, 「NCS 국가 직무능력표준」사이트([www.ncs.go.kr](http://www.ncs.go.kr))에서 색인(찾아보기) 중 하나로 이용할 수 있습니다.

##### 핵심 용어

프리에지, 니퍼, 퓨셔, 폴리시, 네일 파일, 스웨어형, 스웨어 오프형, 라운드형, 오발형, 포인트형

### 3. NCS 학습모듈의 내용 체계



- NCS 학습모듈의 내용은 크게 **학습**, **학습 내용**, **교수·학습 방법**, **평가**로 구성되어 있습니다.

#### 학습

해당 NCS 능력단위요소 명칭을 사용하여 제시한 것입니다.

학습은 크게 학습 내용, 교수·학습 방법, 평가로 구성되며 해당 NCS 능력단위의 능력단위 요소별 지식, 기술, 태도 등을 토대로 학습 내용을 제시한 것입니다.

#### 학습 내용

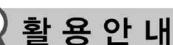
학습 내용은 학습 목표, 필요 지식, 수행 내용으로 구성하였으며, 수행 내용은 재료·자료, 기기(장비·공구), 안전·유의 사항, 수행 순서, 수행 tip으로 구성한 것입니다. 학습모듈의 학습 내용은 업무의 표준화된 프로세스에 기반을 두고 실제 산업현장에서 이루어지는 업무활동을 다양한 방식으로 반영한 것입니다.

#### 교수·학습 방법

학습 목표를 성취하기 위한 교수자와 학습자 간, 학습자와 학습자 간의 상호작용이 활발하게 일어날 수 있도록 교수자의 활동 및 교수 전략, 학습자의 활동을 제시한 것입니다.

#### 평가

평가는 해당 학습모듈의 학습 정도를 확인할 수 있는 평가 준거, 평가 방법, 평가 결과의 피드백 방법을 제시한 것입니다.



예시 : 네일미용 세분류의 ‘네일 기본관리’ 학습모듈의 내용

학습 1	프리에지 형태 만들기(LM1201010403_14v2.1)
학습 2	큐티클 정리하기(LM1201010403_14v2.2)
<b>학습 3 컬러링하기(LM1201010403_14v2.3)</b>	
학습 4	보습제 도포하기(LM1201010403_14v2.4)
학습 5	네일 기본관리 미무리하기(LM1201010403_14v2.5)

#### 학습은

해당 NCS 능력단위요소 명칭을 사용하여 제시하였습니다. 학습은 일반교과의 ‘대단원’에 해당되며, 모듈을 구성하는 가장 큰 단위가 됩니다. 또한 완성된 직무를 수행하기 위한 가장 기초적인 단위로 사용할 수 있습니다.

#### 학습내용은

요소 별 수행준거를 기준으로 제시하였습니다. 일반교과의 ‘중단원’에 해당합니다.

#### 학습목표는

모듈 내의 학습내용을 이수했을 때 학습자가 보여줄 수 있는 행동수준을 의미합니다. 따라서 일반 수업시간의 과목목표로 활용할 수 있습니다.

#### 3-1. 컬러링 매뉴얼 이해

##### 학습목표

- 고객의 요구에 따라 네일 폴리시 색상의 침착을 막기 위한 베이스코트를 아주 얇게 도포할 수 있다.
- 작업 매뉴얼에 따라 네일 폴리시를 일plex 없이 균일하게 도포할 수 있다.
- 작업 매뉴얼에 따라 네일 폴리시 도포 후 컬러 보호와 광택 부여를 위한 톱코트를 바를 수 있다.

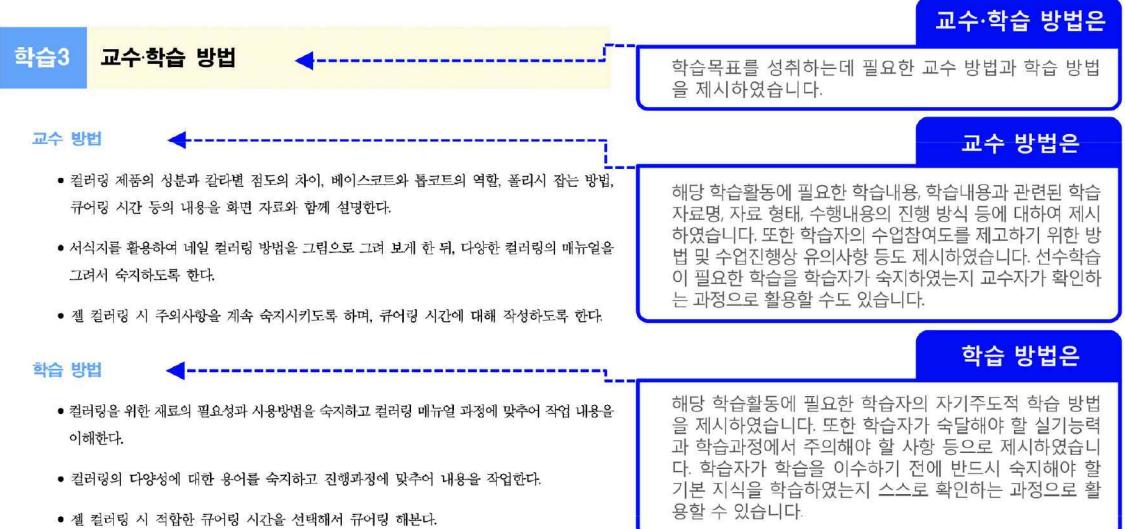
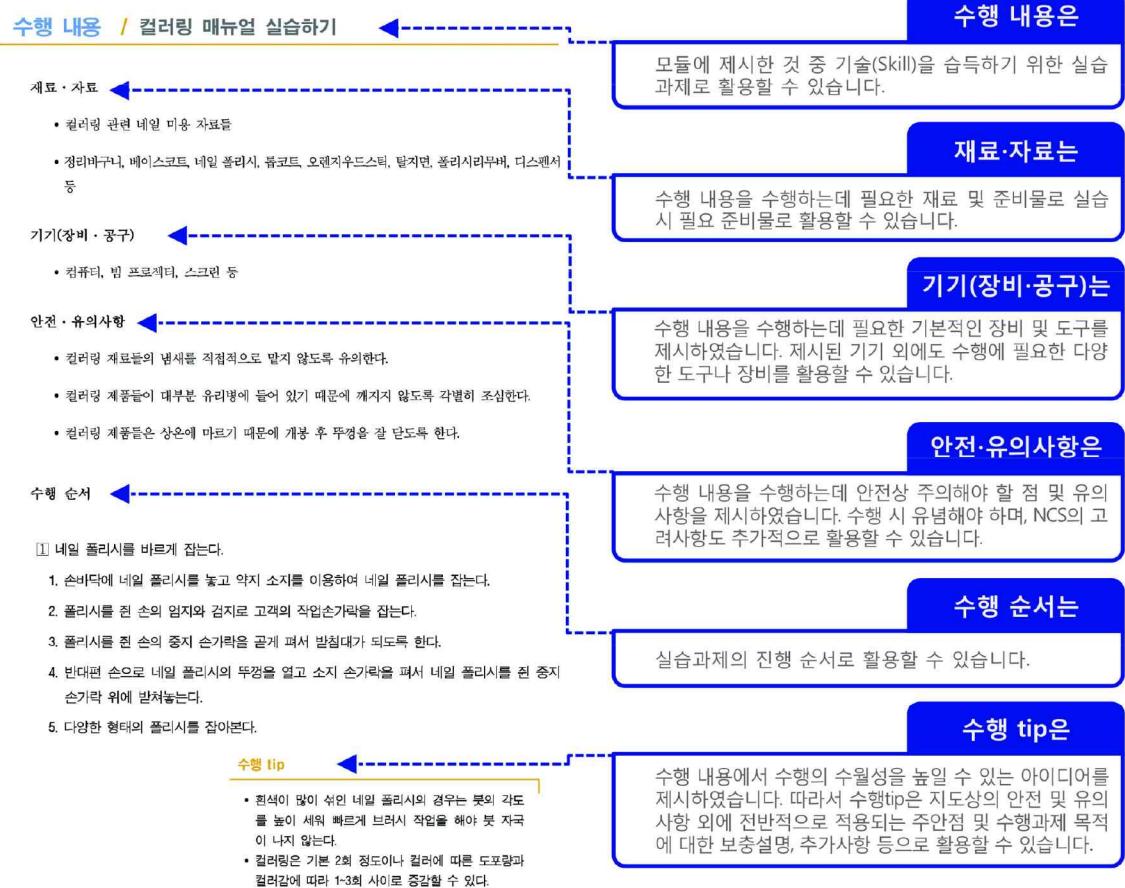
##### 필요 지식 /

##### ① 컬러링 매뉴얼

컬러링 작업 전, 아세톤 또는 네일 폴리시 리무버를 사용하여 손톱표면과 큐티클 주변, 손톱 일부 부분까지 깨끗하게 유분기를 제거해야 한다. 컬러링의 순서는 Base coating 1회 → Polishing 2회 → 컬러수정 → Top coating 1회 → 최종수정의 순서로 한다. 베이스코트는 착색을 방지하고 빌릴성 향상을 위해 가장 먼저 도포하며 컬러링의 마지막에 컬러의 유지와 광택을 위해 톱코트를 도포한다. 네일 보강제(Nail Strengthener)를 바를 시에는 베이스코트를 도포하기 전에 사용한다.

#### 필요지식은

해당 NCS의 지식을 토대로 해당 학습에 대한 이해와 성과를 높이기 위해 알아야 할 주요 지식을 제시하였습니다. 필요지식은 수행에 꼭 필요한 핵심 내용을 위주로 제시하여 교수자의 역할이 매우 중요하며, 이후 수행순서 내용과 연계하여 교수·학습으로 진행할 수 있습니다.



## 학습3 평가

## 평가 준거

- 평가는 학습자가 학습 목표를 성공적으로 달성하였는지를 평가해야 한다.

- 평가는 다음 사항을 평가해야 한다.

학습내용	평가 항목	성취수준
		상 중 하
컬러링 매뉴얼 이해	- 고객의 요구에 따라 네일 폴리시 색상의 질적을 막기 위한 베이스코트를 아주 얇게 도포할 수 있다.	
	- 작업 매뉴얼에 따라 네일 폴리시를 얼룩 없이 균일하게 도포할 수 있다.	
	- 작업 매뉴얼에 따라 네일 폴리시 도포 후 컬러 보호 와 갈색 부여를 위한 들크트를 바를 수 있다.	

## 평가 방법

- 작업장 평가

학습내용	평가 항목	성취수준
		상 중 하
컬러링 매뉴얼 이해	- 고객의 요구에 따라 네일 폴리시 색상의 질적을 막기 위한 베이스코트를 아주 얇게 도포할 수 있다.	
	- 작업 매뉴얼에 따라 네일 폴리시를 얼룩 없이 균일하게 도포할 수 있다.	
	- 작업 매뉴얼에 따라 네일 폴리시 도포 후 컬러 보호 와 갈색 부여를 위한 들크트를 바를 수 있다.	

## 피드백

## 1. 작업장 평가

- 작업 결과물을 확인하여 수정사항을 제시하고 수정 부분을 인지하도록 한다.

## 4. 참고 자료

## 참고 자료

• 김미원(2011).『Nail Study』. 서울: 사)한국네일자격서비스협회.

• 민방정(2015).『비용사(네일)평가』. 서울: 예문사.

• 박온주(2014).『네일미용』. 서울: 정담미디어.

## 5. 활용 서식/부록

## 활용서식

## 프리에지 형태 실습지

1. 프리에지 형태의 이해

모양	이름	특징
	( ) Square nail	강한 느낌의 사각형태 네일의 양끝 모서리 부분이 90° 사각의 형태이다. ( ) 발톱의 형태 활용 내인성 발톱의 보정시에 적용

## 평가는

해당 NCS 능력단위 평가방법과 평가 시 고려 사항을 준용하여 작성하였습니다. 교수자 및 학습자가 평가항목 별 성취수준을 확인하는데 활용할 수 있습니다.

## 평가 준거는

학습자가 해당 학습을 어느 정도 성취하였는지를 평가하기 위한 기준을 제시하고 있습니다. 학습목표와 연계하여 단위수업 시간에 평가항목 별 성취수준을 평가하는데 활용할 수 있습니다.

## 평가 방법은

NCS 능력단위의 평가방법을 준용하였으며, 평가 준거에 따른 평가방법을 2개 이상 제시하였습니다. 평가방법으로는 포트폴리오, 문제해결 시나리오, 서술형 시험, 논술형 시험, 사례연구, 평가자 체크리스트, 작업장 평가 등이 있으며, NCS의 능력단위 요소 별 수행 수준을 평가하는데 가장 적절한 방법을 선정하여 활용할 수 있습니다.

## 피드백은

평가 후에 학습자들에게 평가 결과를 피드백하여 부족한 부분을 알려주고, 학습 결과가 미진한 경우, 해당 부분을 다시 학습하여 학습목표를 달성하는 데 활용할 수 있습니다.

## 참고자료는

해당 학습모듈의 필요지식에 대한 출처와 인용한 참고자료 및 사이트를 제시하였습니다.

## 활용서식은

평가 서식, 실습시트 등 교수학습 시 활용 가능한 다양한 서식들로 구성하였습니다. 과제 진행에서 평가에 이르기까지 필요한 서식을 해당 학습모듈의 특성에 맞춰 개발하거나 기존의 양식을 활용하여 제시하였습니다.

## 부록

## 네일 기본관리 도구와 재료 목록

목록	비고	준비
위생가운	흰색	작업자 착용
위생 마스트	흰색	작업자 착용
보호안경	투명한 렌즈 (안경으로 대체 가능)	작업자 착용
세로정리합	제질, 색상 무관	작업대

## 부록은

활용서식 이외에 교수학습과정에서 참고할 수 있는 자료가 있는 경우 제시하였습니다.

## [NCS-학습모듈의 위치]

대분류	정보 통신
중분류	정보 기술
소분류	정보 기술 개발

세분류	능력단위	학습모듈명
SW아키텍처	요구사항 확인	요구사항 확인
응용SW 엔지니어링	데이터 입출력 구현	데이터 입출력 구현
임베디드SW 엔지니어링	통합 구현	통합 구현
DB엔지니어링	정보시스템 이행	정보시스템 이행
NW엔지니어링	제품소프트웨어 패키징	제품소프트웨어 패키징
보안엔지니어링	서버프로그램 구현	서버프로그램 구현
UI/UX엔지니어링	인터페이스 구현	인터페이스 구현
시스템SW 엔지니어링	애플리케이션 배포	애플리케이션 배포
빅데이터 플랫폼구축	프로그래밍 언어 활용	프로그래밍 언어 활용
핀테크 엔지니어링	응용 SW 기초 기술 활용	응용 SW 기초 기술 활용
데이터아키텍트	애플리케이션 리팩토링	애플리케이션 리팩토링
	인터페이스 설계	인터페이스 설계
	애플리케이션 요구사항 분석	애플리케이션 요구사항 분석
	기능 모델링	기능 모델링

애플리케이션 설계	애플리케이션 설계
정적모델 설계	정적모델 설계
동적모델 설계	동적모델 설계
화면 설계	화면 설계
화면 구현	화면 구현
<b>애플리케이션 테스트 관리</b>	<b>애플리케이션 테스트 관리</b>
애플리케이션 테스트 수행	애플리케이션 테스트 수행
소프트웨어공학 활용	소프트웨어공학 활용
소프트웨어개발 방법론 활용	소프트웨어개발 방법론 활용



---

# 차 례

---

## 학습모듈의 개요

### 학습 1. 애플리케이션 테스트 케이스 설계하기

1-1. 애플리케이션 테스트 케이스 작성	3
1-2. 애플리케이션 테스트 시나리오 작성	19
• 교수 · 학습 방법	29
• 평가	30

### 학습 2. 애플리케이션 통합 테스트하기

2-1. 애플리케이션 통합 테스트 수행	32
2-2. 애플리케이션 테스트 결과 분석	46
2-3. 애플리케이션 개선 조치사항 작성	53
• 교수 · 학습 방법	59
• 평가	60

### 학습 3. 애플리케이션 성능 개선하기

3-1. 애플리케이션 성능 분석	62
3-2. 애플리케이션 성능 개선	72
• 교수 · 학습 방법	85
• 평가	86

참고 자료	88
-------	----

활용 서식	89
-------	----



# 애플리케이션 테스트 관리 학습모듈의 개요

## 학습모듈의 목표

요구사항대로 응용 소프트웨어가 구현되었는지를 검증하기 위해서 테스트 케이스를 작성하고 개발자 통합 테스트를 수행하여 애플리케이션의 성능을 개선할 수 있다.

## 선수학습

소프트웨어공학, 요구사항 확인(2001020201\_16v3), 응용 SW 기초 기술 활용(2001020216\_15v3), 애플리케이션 테스트 수행(2001020227\_16v4)

## 학습모듈의 내용체계

학습	학습 내용	NCS 능력단위 요소	
		코드번호	요소 명칭
1. 애플리케이션 테스트 케이스 설계하기	1-1. 애플리케이션 테스트 케이스 작성 1-2. 애플리케이션 테스트 시나리오 작성	2001020226_16v4.1	애플리케이션 테스트 케이스 설계하기
2. 애플리케이션 통합 테스트하기	2-1. 애플리케이션 통합 테스트 수행 2-2. 애플리케이션 테스트 결과 분석 2-3. 애플리케이션 개선 조치사항 작성	2001020226_16v4.2	애플리케이션 통합 테스트하기
3. 애플리케이션 성능 개선하기	3-1. 애플리케이션 성능 분석 3-2. 애플리케이션 성능 개선	2001020226_16v4.3	애플리케이션 성능 개선하기

## 핵심 용어

테스트 요구사항, 테스트 계획, 테스트 케이스, 테스트 시나리오, 테스트 환경 구축, 통합 테스트, 기능 테스트, 비기능 테스트, 성능 테스트, 테스트 관리



## 학습 1

# 애플리케이션 테스트 케이스 설계하기

학습 2

애플리케이션 통합 테스트하기

학습 3

애플리케이션 성능 개선하기

## 1-1. 애플리케이션 테스트 케이스 작성

### 학습 목표

- 개발하고자 하는 응용 소프트웨어의 특성을 반영한 테스트 방식, 대상과 범위를 결정하여 테스트 케이스를 작성할 수 있다.

### 필요 지식 /

#### ① 응용 소프트웨어의 유형 및 특성 이해

응용 소프트웨어는 불특정 일반인에게 필요한 공통의 기능을 제공하는 상용 소프트웨어와 특정 사용자의 요구사항을 구현하기 위한 서비스 제공 소프트웨어로 구분할 수 있다.

##### 1. 상용 소프트웨어의 특성 및 유형

상업적 목적이나 판매를 목적으로 생산되나, 홍보를 위한 무료 소프트웨어도 포함할 수 있으며, 산업의 특성에 따라 산업 범용 소프트웨어와 산업 특화 소프트웨어로 구분된다.

###### (1) 산업 범용 소프트웨어

산업 범용 소프트웨어는 다시 시스템 소프트웨어, 미들웨어, 응용 소프트웨어로 구분 할 수 있으며, 임베디드/리얼타임 운영체제, 웹 애플리케이션 서버, 영상 관련 소프트웨어 등이 대표적인 유형이다.

###### (가) 시스템 소프트웨어

운영체제(임베디드/리얼타임, 모바일, PC/서버 등), DBMS(DataBase Management System), 데이터 통합, 프로그래밍 언어, 스토리지 소프트웨어, 소프트웨어공학 도구, 가상화 소프트웨어, 시스템 보안 소프트웨어 등이 있다.

###### (나) 미들웨어

웹 애플리케이션 서버, 실시간 데이터 처리, 연계 통합 솔루션, 분산 병렬 처리, 네트워크 관리, 시스템 관리, 클라우드 서비스, 접근 제어 소프트웨어 등이 있다.

#### (다) 응용 소프트웨어

영상 인식/분석, 영상 코덱/스트리밍, 영상 저작/편집/합성, 3D 스캐닝/프린팅, 가상 시뮬레이션, 콘텐츠 보호/관리/유통, 정보검색, 음성 처리, 오피스웨어 소프트웨어 등이 있다.

#### (2) 산업 특화 소프트웨어

특정한 산업 분야에서 요구하는 기능만을 구현하기 위한 목적의 소프트웨어로 자동차, 항공, 조선, 건설, 패션 의류, 농업, 의료, 국방, 공공 분야 등을 지원하는 소프트웨어가 존재한다.

### 2. 서비스 제공 소프트웨어의 특성 및 유형

개발된 소프트웨어의 판매가 아닌 특정한 사용자의 요구사항만을 구현함을 목적으로 생산되며, 신규 기능 개발, 기존에 개발된 기능 개선, 사용자 요구 기능의 추가 개발, 구현된 시스템의 통합을 위한 소프트웨어의 개발 등으로 유형을 분류할 수 있다.

#### (1) 신규 개발 소프트웨어

새로운 서비스 제공을 목적으로 개발되며, 일반적으로 초기 개발 단계, 확장 단계, 기능 고도화 단계 등으로 진행된다.

#### (2) 기능 개선 소프트웨어

기존 서비스 기능에서 사용자 편의성 개선, 응답 속도 개선, 화면 UI(User Interface) 개선, 업무 프로세스 개선 등의 목적으로 개발되는 소프트웨어이다.

#### (3) 추가 개발 소프트웨어

기존 서비스 제공 시스템에 업무 환경의 변화, 산업 환경의 변화, 법/제도의 개정 등으로 인해 새로운 기능을 추가로 개발하는 소프트웨어를 말한다.

#### (4) 시스템 통합 소프트웨어

각각 별도로 서비스되는 시스템을 원스톱(One-Stop) 서비스 제공을 위해 업무 기능 및 데이터 등을 통합하여 개발하는 소프트웨어이다.

## ② 소프트웨어 테스트의 이해

### 1. 소프트웨어 테스트의 개념

소프트웨어 테스트란 구현된 응용 애플리케이션이나 시스템이 사용자가 요구하는 기능의 동작과 성능, 사용성, 안정성 등을 만족하는지 확인하기 위하여 소프트웨어의 결함을 찾아내는 활동이다.

### 2. 소프트웨어 테스트의 필요성

#### (1) 오류 발견 관점

프로그램에 잠재된 오류를 발견하고 이를 수정하여 올바른 프로그램을 개발하는 활동이다.

### (2) 오류 예방 관점

프로그램 실행 전에 코드 리뷰, 동료 검토, 인스펙션 등을 통해 오류를 사전에 발견하는 예방 차원의 활동이다.

### (3) 품질 향상 관점

사용자의 요구사항 및 기대 수준을 만족하도록 반복적인 테스트를 거쳐 제품의 신뢰도를 향상하는 품질 보증 활동이다.

## 3. 소프트웨어 테스트의 기본 원칙

### (1) 소프트웨어 테스트의 원리

(가) 테스팅은 결함이 존재함을 밝히는 활동이다.

테스팅은 소프트웨어의 잠재적인 결함을 줄일 수 있지만, 결함이 발견되지 않아도 결함이 없다고 증명할 수 없음을 나타낸다.

(나) 완벽한 테스팅은 불가능하다.

무한 경로, 무한 입력 값, 무한 시간이 소요되어 완벽하게 테스트할 수 없으므로 리스크 분석과 우선순위를 토대로 테스트에 집중할 것을 의미한다.

(다) 테스팅은 개발 초기에 시작해야 한다.

애플리케이션의 개발 단계에 테스트를 계획하고 SDLC(Software Development Life Cycle)의 각 단계에 맞춰 전략적으로 접근하는 것을 고려하라는 뜻이다.

(라) 결함 집중(Defect Clustering)

애플리케이션 결함의 대부분은 소수의 특정한 모듈에 집중되어 존재한다.

(마) 살충제 패러독스(Pesticide Paradox)

동일한 테스트 케이스로 반복 실행하면 결함을 발견할 수 없으므로 주기적으로 테스트 케이스를 리뷰하고 개선해야 한다.

(바) 테스팅은 정황(Context)에 의존한다.

정황과 비즈니스 도메인에 따라 테스트를 다르게 수행하여야 한다.

(사) 오류-부재의 궤변(Absence of Errors Fallacy)

사용자의 요구사항을 만족하지 못하는 오류를 발견하고 그 오류를 제거하였다 해도, 해당 애플리케이션의 품질이 높다고 말할 수 없다.

### (2) 소프트웨어 테스트 프로세스

일반적인 테스트 프로세스는 테스트 계획, 테스트 분석, 테스트 디자인, 테스트 케이스 및 시나리오 작성, 테스트 수행, 테스트 결과 평가 및 리포팅의 절차로 이루어진다.



[그림 1-1] 테스트 프로세스

### (3) 소프트웨어 테스트 산출물

#### (가) 테스트 계획서

테스트 목적과 범위 정의, 대상 시스템 구조 파악, 테스트 수행 절차, 테스트 일정, 조직의 역할 및 책임 정의, 종료 조건 정의 등 테스트 수행을 계획한 문서

#### (나) 테스트 케이스

테스트를 위한 설계 산출물로, 응용 소프트웨어가 사용자의 요구사항을 준수하는지 확인하기 위해 설계된 입력 값, 실행 조건, 기대 결과로 구성된 테스트 항목의 명세서

#### (다) 테스트 시나리오

테스트 수행을 위한 여러 개의 테스트 케이스의 집합으로 테스트 케이스의 동작 순서를 기술한 문서이며, 테스트를 위한 절차를 명세한 문서

#### (라) 테스트 결과서

테스트 결과를 정리한 문서로 테스트 프로세스를 리뷰하고, 테스트 결과를 평가하고 리포팅하는 문서

## ③ 소프트웨어 테스트의 유형

### 1. 프로그램 실행 여부

#### (1) 정적 테스트

프로그램 실행 없이 소스 코드의 구조를 분석하여 논리적으로 검증하는 테스트로 인스펙션, 코드 검사, 워크스루 등이 있다.

#### (2) 동적 테스트

프로그램의 실행을 요구하는 테스트로 화이트박스 테스트와 블랙박스 테스트가 있다.

## 2. 테스트 기법

### (1) 화이트박스 테스트

프로그램의 내부 로직(수행 경로 구조, 루프 등)을 보면서 테스트를 수행한다.

### (2) 블랙박스 테스트

프로그램의 외부 사용자 요구사항 명세를 보면서 테스트, 주로 구현된 기능을 테스트 한다.

## 3. 테스트에 대한 시각

### (1) 검증(Verification)

제품의 생산 과정을 테스트한다(Are we building the product right?).

올바른 제품을 생산하고 있는지 검증하는 것을 의미한다.

### (2) 확인(Validation)

생산된 제품의 결과를 테스트한다(Are we building the right product?)

생산된 제품이 정상적으로 동작하는지 확인하는 것을 의미한다.

## 4. 테스트 목적

### (1) 회복(Recovery) 테스트

시스템에 고의로 실패를 유도하고 시스템이 정상적으로 복귀하는지 테스트한다.

### (2) 안전(Security) 테스트

불법적인 소프트웨어가 접근하여 시스템을 파괴하지 못하도록 소스코드 내의 보안적인 결함을 미리 점검하는 테스트이다.

### (3) 강도(Stress) 테스트

시스템에 과다 정보량을 부과하여 과부하 시에도 시스템이 정상적으로 작동되는지를 검증하는 테스트이다.

### (4) 성능(Performance) 테스트

사용자의 이벤트에 시스템이 응답하는 시간, 특정 시간 내에 처리하는 업무량, 사용자 요구에 시스템이 반응하는 속도 등을 테스트한다.

### (5) 구조(Structure) 테스트

시스템의 내부 논리 경로, 소스코드의 복잡도를 평가하는 테스트이다.

### (6) 회귀(Regression) 테스트

변경 또는 수정된 코드에 대하여 새로운 결함 발견 여부를 평가하는 테스트이다.

### (7) 병행(Parallel) 테스트

변경된 시스템과 기존 시스템에 동일한 데이터를 입력 후 결과를 비교하는 테스트이다.

## 5. 테스트 종류

### (1) 명세 기반 테스트

주어진 명세를 빠짐없이 테스트 케이스로 구현하고 있는지 확인하는 테스트를 말하며, 동등 분할, 경계 값 분석, 유한상태 기계기반, 결정 테이블, 정형명세 기반, 유스케이스, 페어와이즈, 직교배열 테스트 등이 있다.

### (2) 구조 기반 테스트

소프트웨어 내부 논리 흐름에 따라 테스트 케이스를 작성하고 확인하는 테스트를 말하며, 구문 기반, 결정 기반, 조건 기반, 조건결정 기반, 변경조건 결정 기반, 멀티조건 기반 커버리지 테스트 등이 있다.

### (3) 경험 기반 테스트

유사 소프트웨어나 유사 기술 평가에서 테스터의 경험을 토대로 한, 직관과 기술 능력을 기반으로 수행하는 테스트를 말한다.

## ④ 테스트 케이스와 테스트 오라클의 이해

### 1. 테스트 케이스의 개념

명세 기반 테스트의 설계 산출물로, 특정한 프로그램의 일부분 또는 경로에 따라 수행하거나, 특정한 요구사항을 준수하는지 확인하기 위해 설계된 입력 값, 실행 조건, 기대 결과로 구성된 테스트 항목의 명세서를 말한다.

### 2. 테스트 케이스 작성

테스트 케이스의 정확성, 재사용성, 간결성 보장을 위해 아래의 절차에 따라 작성한다.

#### (1) 테스트 계획 검토 및 자료 확보

테스트 대상 프로젝트 범위와 접근 방법 이해를 위하여 테스트 계획을 재검토한다.

테스트 대상 시스템 자료와 정보를 확보하여, 시스템 요구사항과 기능 명세서를 검토한다.

#### (2) 위험 평가 및 우선순위 결정

결합 해결에 있어 상대적 중요성을 지니며 테스트의 초점을 결정한다.

#### (3) 테스트 요구사항 정의

시스템 요구사항, 테스트 대상 재검토, 테스트할 특성, 조건, 기능을 식별 및 분석한다.

#### (4) 테스트 구조 설계 및 테스트 방법 결정

테스트 케이스의 일반적 형식을 결정하고, 테스트 케이스 분류 방법을 결정한다.

테스트 절차, 장비, 도구, 테스트 문서화 방법을 결정한다.

#### (5) 테스트 케이스 정의

각 요구사항에 대해 테스트 케이스를 작성하고, 입력 값, 실행 조건, 예상 결과를 기술한다.

## (6) 테스트 케이스 타당성 확인 및 유지 보수

기능 또는 환경 변화에 따라 테스트 케이스를 갱신하고, 테스트 케이스의 유용성을 검토한다.

### 3. 테스트 오라클의 개념

#### (1) 테스트 오라클 정의

테스트의 결과가 참인지 거짓인지를 판단하기 위해서 사전에 정의된 참 값을 입력하여 비교하는 기법 및 활동을 말한다.

#### (2) 테스트 오라클 유형

##### (가) 참(True) 오라클

모든 입력 값에 대하여 기대하는 결과를 생성함으로써 발생된 오류를 모두 검출할 수 있는 오라클이다.

##### (나) 샘플링(Sampling) 오라클

특정한 몇 개의 입력 값에 대해서만 기대하는 결과를 제공해 주는 오라클이다.

##### (다) 휴리스틱(Heuristic) 오라클

샘플링 오라클을 개선한 오라클로, 특정 입력 값에 대해 올바른 결과를 제공하고, 나머지 값들에 대해서는 휴리스틱(추정)으로 처리하는 오라클이다.

##### (라) 일관성 검사(Consistent) 오라클

애플리케이션 변경이 있을 때, 수행 전과 후의 결과 값이 동일한지 확인하는 오라클이다.

#### (3) 오라클 적용 방안

참 오라클은 주로 항공기, 임베디드, 발전소 소프트웨어 등 미션 크리티컬한 업무에 적용하고, 샘플링/추정 오라클은 일반, 업무용, 게임, 오락 등의 일반적인 업무에 적용한다.

## 수행 내용 / 애플리케이션 테스트 케이스 작성하기

---

### 재료 · 자료

- SW 개발 국제 표준(ISO/IEC 9126, ISO/IEC 14598, ISO15504, ISO/IEC 12207, CMMI 등)
- 소프트웨어 테스트 관련 국제 표준(ISO/IEC 29119)
- 제안 요청서, 요구사항 정의서, 테스트 계획서

### 기기(장비 · 공구)

- 전산 장비(컴퓨터, 프린터, 빔 프로젝터, 네트워크 장비 등)
- OA 도구(워드프로세서, 스프레드시트, 프레젠테이션 도구 등)
- CASE 도구(SW설계 지원 도구, SW 개발 지원 도구, SW 테스트 지원 도구 등)

### 안전 · 유의 사항

- 테스트 실습은 정해진 방법과 절차에 따라 실시하여야 한다.
- 테스트 지원 도구의 사용 방법과 유의 사항에 대해 습득하여야 한다.
- 응용 소프트웨어의 특성을 반영한 테스트 방법을 정의하여야 한다.
- 테스트 요구사항 분석을 기반으로 테스트 케이스를 작성하여야 한다.
- 테스트 목표에 맞는 테스트 케이스를 작성하여야 한다.

### 수행 순서

① 개발하고자 하는 응용 소프트웨어의 유형을 분류하고, 유형별 특성을 정리한다.

1. 응용 소프트웨어의 사용 대상과 제공 기능을 조사하여 유형을 분류하고, 유형별 특성을 반영한 테스트 시 중점 사항에 대해 정리한다.
  - (1) 상용 소프트웨어와 서비스 제공 소프트웨어를 분류한다.
  - (2) 산업 범용 응용 소프트웨어와 산업 특화 응용 소프트웨어를 분류한다.
  - (3) 응용 소프트웨어의 개발 환경 및 사용 환경을 분류한다.
  - (4) 서비스 제공 응용 소프트웨어의 유형별 특성을 정리한다.

<표 1-1> 응용 소프트웨어 유형별 특성 정리 예시

소프트웨어 명	제공 유형	기능 유형	사용 환경	개발 유형	중점 사항	...
A. xx오픈DB 구축	서비스 제공 소프트웨어	산업 특화	Web	신규 개발	기능 구현 시 사용자 요구사항의 누락 여부	
B. xx통합서비스 구현	서비스 제공 소프트웨어	산업 특화	Web	시스템 통합	기존 시스템과 신규 시스템의 데이터 손실 및 정합성 여부	
C. xx오피스	상용 소프트웨어	산업 범용	C/S	신규 개발	다양한 OS환경 지원 여부	

## ② 응용 소프트웨어의 요구사항을 분석한다.

요구사항 명세서에 반영된 사용자 요구사항, 시스템 요구사항, 애플리케이션의 기능 요구 사항 및 비기능 요구사항에 대해 분석한다.

<표 1-2> 응용 소프트웨어 요구사항 목록 예시

요구사항 ID	요구사항 명	요구사항 내용	유형	중요도	난이도	제약사항	출처	...	...
Ra-XXX-012	관리자	통합계시 판 관리 기능 기능 구현	기능	상	5		SFR-001		
Ra-XXX-013	쪽지 송수신 기능구현	웹UI를 통한 쪽지... 수신자별, 그룹별, 개인별 쪽지 발송... ...사용자의 질의요청	기능	상	5	파일첨부 기능 삭제	SFR-001		
Ra-XXX-031	평균응답 시간	에 대한 결과 페이지를 4초 이내 출력	비기능 (성능)	상	4		PER-002		

1. 기능 요구사항 명세를 분석하고 분석표를 작성한다.

사용자 요구사항의 기능 구현 여부를 확인할 수 있도록 구현 방안과 검증 방안을 포함한 기능 요구사항 분석표를 작성한다.

- (1) 요구사항 명세서의 업무 기능에 대한 사용자 요구사항을 확인한다.
- (2) 애플리케이션의 기능 구현 방안 및 검증 방안을 확인한다.
- (3) 기능 요구사항 명세에 대한 분석표를 작성한다.

<표 1-3> 애플리케이션의 기능 요구사항 분석표 예시

요구사항ID	요구사항 명	요구사항 내용	구현 방안	검증 방안	테스트ID
FR-007	문화xxDB관리시스템 구축	문화xxDB 서비스 및 콘텐츠 관리를 위한 웹 기반 관리페이지 구축	-웹 기반 관리 페이지 구축 -모든 내용을 한눈에 볼 수 있도록 제작	웹 기반 관리 페이지가 정상적으로 구축됨을 구현된 기능에 대해 단위 테스트와 통합 테스트를 수행하여 검증	UT-DM-001 TC-DM-001
FR-008	문화데이터 콘텐츠 관리	문화데이터 콘텐츠의 등록/수정/삭제 /조회 기능 구현	문화xx시스템에서 유형별로 정리된 DB를 토대로 메뉴 구성	기능 구현 여부 확인 -문화xx시스템 DB와 구현된 메뉴의 일치성 여부 확인	UT-DM-002 TC-DM-002
...	...	...	...	...	...
FR-011	기관연계 관리	xx시스템에서 정보 변경 시 oo시스템에 변경 내역을 푸시(push)하는 라이브러리 제공	REST 기반으로 변경 내역을 푸시(push)하는 기능을 수행하는 라이브러리 개발	xx기관의 oo시스템으로 구축 예정	UT-DM-002 TC-DM-002

2. 비기능 요구사항 명세를 분석하고 분석표를 작성한다.

애플리케이션의 비기능 요구사항(보안, 성능, 품질, 테스트 등)에 대해 확인하고, 검증 방안을 포함한 분석표를 작성한다.

- (1) 요구사항 명세서의 비기능 요구사항 확인 및 검증 방안을 분석한다.
- (2) 애플리케이션의 비기능 검증 방안을 확인한다.
- (3) 비기능 요구사항 명세에 대한 분석표를 작성한다.

<표 1-4> 애플리케이션의 비기능 요구사항 분석표 예시

요구사항ID	요구사항 명	요구사항 내용	구현 방안	검증 방안	산출물
xxPER-001	시스템성능	관련 서버 및 SW의 성능을 테스트하고 결과 보고, CPU사용률은 70%이하	구축된 시스템의 성능 테스트 결과 제시	시스템 테스트 결과서	
xxPER-002	평균응답 시간	정상 상태에서 사용자의 질의요청에 결과 페이지를 4초 이내에 출력	구축된 시스템의 성능 테스트 결과 제시	시스템 테스트 결과서	
...	...	...	...	...	...
xxPER-005	시스템 처리량	초당 최소 100건의 사용자 질의 처리, 최대 부하 상태에서 초당 50건의 질의 처리	구축된 시스템의 성능 테스트 결과 제시	시스템 테스트 결과서	

### ③ 분석된 요구사항에 근거한 테스트 방식, 대상과 범위를 결정한다.

개발하고자 하는 응용 소프트웨어의 특성을 반영한 테스트 방식과 테스트 대상과 테스트 범위를 결정한다.

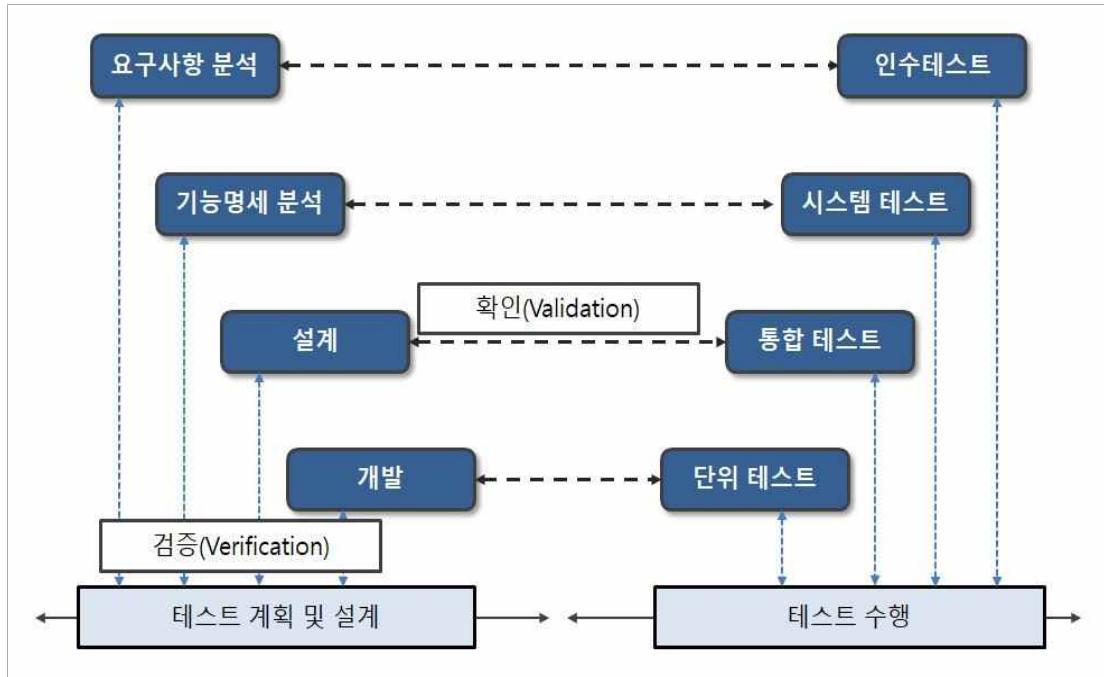
#### 1. 애플리케이션의 기능 구현 여부 및 비기능 검증을 위한 테스트 방식을 결정한다.

개발된 애플리케이션의 코드 실행 여부, 프로그램 내부 로직 검토 여부, 프로그램 외부 명세 참조 여부, 프로그램 실행 여부 등 다양한 테스트 방식 중 최적의 테스트 방식을 결정 한다.

##### (1) 테스트 방식을 결정하는 요소를 식별한다.

###### (가) 진행 중인 프로젝트의 개발 수명 주기와 테스트 단계를 파악한다.

요구사항 분석, 기능 명세 분석, 테스트 설계, 테스트 개발 등의 테스트 계획 및 설계 단계와, 단위 테스트, 통합 테스트, 시스템 테스트, 인수 테스트라는 일련의 애플리케이션 테스트 수행 단계의 관계를 파악한다.



[그림 1-2] V-모델과 테스트 단계

(나) 각 테스트의 단계별 기능 검증 사항에 대해 파악한다.

1) 단위 테스트 단계의 기능 검증 사항을 파악한다.

구현된 모듈(함수, 서브루틴, 컴포넌트 등)의 기능 수행 여부를 판정하고, 내부에 존재하는 논리적 오류를 검출할 수 있는 방안을 파악한다.

2) 통합 테스트 단계의 기능 검증 사항을 파악한다.

모듈 간의 인터페이스 연계를 검증하고, 모듈 간의 인터페이스 오류를 확인하고, 모듈 간의 상호 작용 및 연계 동작 여부를 판정하는 방안을 파악한다.

3) 시스템 테스트 단계의 기능 검증 사항을 파악한다.

단위, 통합 테스트 후 전체 시스템이 정상적으로 작동하는지 판정하는 기능 명세를 확인하는 방안을 파악한다.

4) 인수 테스트 단계의 기능 검증 사항을 파악한다.

사용자가 요구분석 명세서에 명시된 사항을 모두 충족하는지 판정하고, 시스템이 예상대로 동작하고 있는지를 판정하는 방안을 파악한다.

(다) 작업 분할 구조도(WBS, Work Breakdown Structure)와 현재 프로젝트 진행 상황을 판단한다.

(라) 테스트 일정과 테스트 투입 인력 등 테스트 비용에 대해 산정한다.

(2) 개발 수명 주기별 기능 구현 여부를 검증할 테스트 방식을 결정한다.

프로젝트 초기 단계에 개발된 코드를 수동 또는 자동화된 기법으로 검토하는 정적 테스트와 프로그램을 실행하며 테스트하는 동적 테스트 중 적절한 방식을 결정한다.

(3) 애플리케이션의 비기능 검증을 위한 테스트 방식을 결정한다.

비기능 요소를 식별하고, 성능 테스트, 보안 테스트, 품질 테스트, 사용성 테스트 등 적절한 테스트 방식을 결정한다.

## 2. 테스트 대상과 테스트 범위를 결정한다.

개발된 응용 소프트웨어가 단위 시스템인지, 통합 시스템인지, 연계 시스템인지 여부를 식별하고, 테스트 대상과 테스트 범위를 결정한다.

(1) 테스트 대상을 결정하는 요소를 식별한다.

(가) 개발된 애플리케이션의 시스템 유형에 따라 테스트 대상을 판단한다.

(나) 단일 시스템 구현과 연계 및 통합 시스템 구현을 판단한다.

(다) 연계 및 통합 시스템의 경우 개발 시스템에서의 테스트 가능 여부를 판단한다.

(2) 테스트 범위를 결정하는 요소를 식별한다.

(가) 개발하고자 하는 애플리케이션의 범위 및 상위 요구사항에 따라 테스트 범위를 판단하다.

(나) 기능을 검증하는 테스트의 경우 추가 사항으로 테스트 횟수를 결정한다.

(다) 비기능 검증을 위한 테스트의 경우 결과 값 산출 방식을 결정한다.

(3) 식별된 테스트 결정 요소를 기반으로 테스트 대상과 테스트 범위를 결정한다.

## ④ 결정된 테스트 방식, 대상과 범위를 고려하여 테스트 케이스를 작성한다.

### 1. 테스트 케이스 작성에 필요한 항목을 결정한다.

(1) 테스트 케이스 항목 중 공통 작성 항목 요소를 식별한다.

(가) 테스트 단계 명, 작성자, 승인자, 작성 일자, 문서 버전을 식별한다.

단위/통합/시스템/인수 테스트 등의 테스트 단계와, 테스트 케이스 작성자, 승인자, 작성 일자, 버전 등을 작성한다.

(나) 대상 시스템을 식별한다.

애플리케이션 개발 서버 또는 개발 시스템 명 등을 작성한다.

(다) 변경 여부를 식별한다.

테스트 케이스 변경 여부 및 변경 사유 등을 작성한다.

(라) 테스트 범위를 식별한다.

테스트 대상 애플리케이션의 기능별 테스트 범위 및 업무별 테스트 범위를 식별한다.

(마) 테스트 조직을 식별한다.

테스트 케이스 작성 및 테스트 수행을 담당할 조직을 식별한다.

(2) 개별 테스트 케이스 항목 요소를 식별한다.

(가) 테스트 ID를 작성한다.

테스트 케이스를 고유하게 식별하기 위한 ID를 작성한다.

(나) 테스트 목적을 작성한다.

테스트 시 고려해야 할 중점 사항이나 테스트 케이스의 목적을 작성한다.

(다) 테스트할 기능을 요약한다.

애플리케이션의 테스트할 기능을 간략하게 작성한다.

(라) 입력 데이터를 작성한다.

테스트 실행 시 입력할 데이터(입력 값, 선택 버튼, 체크리스트 값 등)를 작성한다.

(마) 기대 결과를 작성한다.

테스트 실행 후 기대되는 결과 데이터(출력 데이터, 결과 화면, 기대 동작 등)를 작성한다.

(바) 테스트 환경을 설정한다.

테스트 시 사용할 물리적, 논리적 테스트 환경, 사용할 데이터, 결과 기록 서버 등 의 내용을 작성한다.

(사) 전제 조건을 설정한다.

테스트 간의 종속성, 테스트 수행 전 실행되어야 할 고려 사항 등을 작성한다.

(아) 성공/실패 기준을 설정한다.

테스트를 거친 애플리케이션 기능의 성공과 실패를 판단하는 조건을 명확하게 작성한다.

(자) 기타 요소를 식별하여 설정한다.

사용자의 테스트 요구사항 중 특별히 고려해야 할 내용을 간략하게 기술한다.

2. 식별된 항목 요소를 반영한 테스트 케이스를 작성한다.

공통 항목 요소와 개별 항목 요소를 적용하여 테스트 케이스 명세를 작성한다.

<표 1-5> 단위 테스트 케이스 명세 작성 예시

테스트 케이스		프로젝트 명 : xx시스템 구축	대상 시스템 명 : xxDB 서비스 시스템
단계 명 : 단위 테스트	작성자 : 홍길동	승인자 : 박승인	변경 : 최초 작성
작성일 : 2017-02-02	버전 : 1.0	테스트 범위 : 단일 시스템	테스트 조직 : A 팀
테스트 ID	UT-DM-xxx	테스트 일자	2017-03-00
테스트 목적	사용자 로그인 테스트		
테스트 기능	사용자의 ID와 패스워드 검증		
입력 데이터	사용자 ID, 패스워드		
테스트 단계	케이스 설명	예상 출력 중 요 력도	확인 비고
	사용자가 ID/PW 없이 로그인을 시도한다.	오류 메시지	
	사용자가 ID만 입력 후 로그인을 시도한다.	오류 메시지	
	사용자가 존재하지 않는 ID(abdc)로 로그인을 시도 한다.	오류 메시지	
	사용자가 잘못된 패스워드 (ID= “dlsung01” ,PW= “pw1234” )로 로그인을 시도한다.	오류 메시지	
테스트 환경	사용자가 올바른 ID와 패스워드 (ID= “dlsung01” ,PW= “PW12345” )로 로그인을 시도한다.	로그인 성공	
		메시지	
전제 조건	개발 환경의 테스트 서버, 형상 관리 서버		
성공/실패 기준	기대 결과가 정상적으로 출력되면 성공		
기타 테스트 의견 사항			

### 수행 tip

- 테스트를 효과적으로 진행하기 위해 테스트 대상을 목적에 맞게 분류하고, 테스트 케이스를 작성하여 무엇을 테스트할 것인지 구체화한다.
- 테스트 케이스를 어떤 단계로 실행해야 하는지를 기술하여 효율적인 테스트가 수행되도록 테스트 절차를 작성한다.

# 1-2. 애플리케이션 테스트 시나리오 작성

## 학습 목표

- 개발하고자 하는 응용 소프트웨어의 특성을 반영한 테스트 방식, 대상과 범위가 적용된 시나리오를 정의할 수 있다.
- 애플리케이션 테스트 수행에 필요한 테스트 데이터, 테스트 시작 및 종료 조건 등을 준비할 수 있다.

## 필요 지식 /

### ① 테스트 시나리오의 이해

#### 1. 테스트 시나리오의 개념

##### (1) 테스트 시나리오의 정의

테스트 수행을 위한 여러 테스트 케이스의 집합으로서, 테스트 케이스의 동작 순서를 기술한 문서이며 테스트를 위한 절차를 명세한 문서이다.

##### (2) 테스트 시나리오의 필요성

테스트 수행 절차를 미리 정함으로써 설계 단계에서 중요시되던 요구사항이나 대안 흐름과 같은 테스트 항목을 빠짐없이 테스트하기 위함이다.

#### 2. 테스트 시나리오 작성 시 유의점

##### (1) 테스트 시나리오 분리 작성

테스트 항목을 하나의 시나리오에 모두 작성하지 않고, 시스템별, 모듈별, 항목별 테스트 시나리오를 분리하여 작성한다.

##### (2) 고객의 요구사항과 설계 문서 등을 토대로 테스트 시나리오를 작성한다.

##### (3) 각 테스트 항목은 식별자 번호, 순서 번호, 테스트 데이터, 테스트 케이스, 예상 결과, 확인 등의 항목을 포함하여 작성한다.

### ② 테스트 환경 구축의 이해

#### 1. 테스트 환경 구축의 개념

개발된 응용 소프트웨어가 실제 운영 시스템에서 정상적으로 작동하는지 테스트할 수 있도록 하기 위하여 실제 운영 시스템과 동일 또는 유사한 사양의 하드웨어, 소프트웨어, 네트워크 등의 시설을 구축하는 활동이다.

## 2. 테스트 환경 구축의 유형

### (1) 하드웨어 기반의 테스트 환경 구축

서버 장비(WAS 서버, DBMS 서버), 클라이언트 장비(노트북 또는 PC), 네트워크(내부 LAN 또는 공용 인터넷 라인) 장비 등의 장비를 설치하는 작업이다.

### (2) 소프트웨어 기반의 테스트 환경 구축

구축된 하드웨어 환경에 테스트할 응용 소프트웨어를 설치하고 필요한 데이터를 구축하는 작업이다.

### (3) 가상 시스템 기반의 테스트 환경 구축

물리적으로 개발 환경 및 운영 환경과 별개로 독립된 테스트 환경을 구축하기 힘든 경우에는, 가상 머신(Virtual Machine) 기반의 서버 또는 클라우드 환경을 이용하여 테스트 환경을 구축하고, 네트워크는 VLAN과 같은 기법을 이용하여 논리적 분할 환경을 구축할 수 있다.

## ③ 테스트 데이터 및 테스트 조건의 이해

### 1. 테스트 데이터의 개념

컴퓨터의 동작이나 시스템의 적합성을 시험하기 위해 특별히 개발된 데이터 집합으로서 프로그램의 기능을 하나씩 순번에 따라 확실하게 테스트할 수 있도록 조건을 갖춘 데이터를 말한다.

#### (1) 테스트 데이터의 필요성

테스트 수행 시 잘못된 데이터를 사용하면 잘못된 결과가 도출되어 시간을 낭비하고 비용만 소진하는 결과가 나온다. 따라서 테스트를 효율적으로 운용하고 데이터의 기밀을 유지하며 신뢰 및 예측 가능한 테스트를 위해 테스트 데이터의 준비가 필요하다.

#### (2) 테스트 데이터의 유형

선행된 연산에 의해 얻어진 실제 데이터와 인위적으로 만들어진 가상의 데이터로 구분된다.

#### (3) 테스트 데이터 준비

실제 데이터는 연산에 의해 준비하거나 실제 운영 데이터를 복제하여 준비할 수 있으며, 가상의 데이터는 스크립트를 통해서 생성할 수 있다.

### 2. 테스트 조건의 개념

#### (1) 테스트 시작 조건

테스트 계획의 수립, 사용자 요구사항에 대한 테스트 명세의 작성, 투입 조직 및 참여 인력의 역할과 책임의 정의, 테스트 일정의 확정, 테스트 환경의 구축 등이 완료되었을 때 테스트를 시작하는 조건이다.

을 때 테스트를 시작하도록 조건을 정의할 수 있으며, 단계별 또는 회차별 테스트 수행을 위해 모든 조건을 만족하지 않아도 테스트를 시작할 수 있다.

### (2) 테스트 종료 조건

테스트의 종료는 정상적인 테스트를 모두 수행한 경우, 차기 일정의 도래로 테스트 일정이 만료되었을 경우, 테스트에 소요되는 비용을 모두 소진한 경우 등 업무 기능의 중요도에 따라 조건을 달리 정할 수 있다.

### (3) 테스트 성공과 실패의 판단 기준

기능 및 비기능 테스트 시나리오에 기술된 예상 결과를 만족하면 성공으로 아니라면 실패로 판단할 수 있으며, 또는 동일한 데이터 또는 이벤트를 중복하여 테스트하여도 여전히 이전 테스트와 같은 결과가 나올 때 성공으로 판단할 수도 있다.

## 수행 내용 / 테스트 시나리오 작성 및 테스트 환경 구축하기

---

### 재료 · 자료

- SW 개발 국제 표준(ISO/IEC 9126, ISO/IEC 14598, ISO/IEC 15504, ISO/IEC 12207, CMMI 등)
- 소프트웨어 테스트 관련 국제 표준(ISO/IEC 29119)
- 요구사항 명세서, 테스트 계획서, 테스트 설계서, 테스트 데이터

### 기기(장비 · 공구)

- 전산 장비(컴퓨터, 프린터, 빔 프로젝터, 네트워크 장비 등)
- OA 도구(워드프로세서, 스프레드시트, 프레젠테이션 도구 등)
- CASE 도구(SW설계 지원 도구, SW 개발 지원 도구, SW 테스트 지원 도구 등)

### 안전 · 유의 사항

- 테스트 지원 도구의 사용 방법과 유의 사항에 대해 습득하여야 한다.
- 응용 소프트웨어의 특성을 반영한 테스트 시나리오를 정의하여야 한다.
- 테스트 방법별 테스트 시나리오를 작성하여야 한다.
- 테스트 수행은 개발 및 실제 운영 환경과는 분리된 시스템으로 구축하여야 한다.

- 테스팅 서버는 운영 서버와 동일하거나 유사한 성능을 보유한 사양으로 준비하여야 한다.
- 테스트 수행을 위한 데이터를 미리 준비하고 검증하여야 한다.

## 수행 순서

### ① 기준에 수립된 테스트 계획을 검토한다.

1. 테스트 목적, 범위 설정 및 테스트 전략을 수립한다.
  - (1) 해당 단계에서의 테스트 목적을 설정한다.
  - (2) 테스트 대상 시스템의 범위를 설정한다.
  - (3) 테스트 대상 애플리케이션 기능의 범위를 설정한다.
  - (4) 수행할 테스트 방법을 결정한다.
  - (5) 대상 시스템에 대한 소프트웨어 및 하드웨어 구조를 파악하여 정리한다.
2. 테스트 일정 계획을 검토한다.
  - (1) 테스트 활동의 주요 일정을 정의한다.
  - (2) 단계별 테스트 투입 인력을 검토한다.

통합테스트 계획서		프로젝트명: xx 시스템 기능개선 사업														
		시스템명 : xx시스템														
문서번호 : CS5xx-TS021	작성자 : 흥길동	작성일 : 2016/10/17														
버전 : 1.0																
<b>목차</b>																
<table> <tr> <td>1. 개요 .....</td><td>4</td></tr> <tr> <td>2. 시험조직 .....</td><td>5</td></tr> <tr> <td>3. 테스트 수행 절차 .....</td><td>6</td></tr> <tr> <td>4. 품질목표 .....</td><td>6</td></tr> <tr> <td>5. 판정 및 재시험 .....</td><td>7</td></tr> <tr> <td>6. 테스트 환경 .....</td><td>8</td></tr> <tr> <td>7. 테스트 세부 일정 .....</td><td>9</td></tr> </table>			1. 개요 .....	4	2. 시험조직 .....	5	3. 테스트 수행 절차 .....	6	4. 품질목표 .....	6	5. 판정 및 재시험 .....	7	6. 테스트 환경 .....	8	7. 테스트 세부 일정 .....	9
1. 개요 .....	4															
2. 시험조직 .....	5															
3. 테스트 수행 절차 .....	6															
4. 품질목표 .....	6															
5. 판정 및 재시험 .....	7															
6. 테스트 환경 .....	8															
7. 테스트 세부 일정 .....	9															

[그림 1-3] 통합 테스트 계획서 목차 예시

② 테스트 조직 및 역할을 정의한다.

1. 각 테스트 참가 조직별 역할 및 책임을 정의한다.

개발자, 테스트 책임자(PL), 프로젝트 책임자(PM), 인수 책임자(사용자 또는 주관 기관)별 책임과 역할을 정의한다.

<표 1-6> 테스트 수행을 위한 업무 분장 작성 예시

역할	책임 및 역할	담당자	비고
인수 책임자 (주관 기관)	<ul style="list-style-type: none"> <li>- 인수 테스트 설계서 작성</li> <li>- 인수 테스트 실시</li> <li>- 인수 테스트 결과 승인</li> </ul>	000	
PM	<ul style="list-style-type: none"> <li>- 테스트 계획 검토 및 승인</li> <li>- 테스트 진척 보고 및 결과 보고 검토</li> <li>- 테스트 수행 통제</li> </ul>	000	
PL	<ul style="list-style-type: none"> <li>- 테스트 계획 수립 및 검토</li> <li>- 테스트 설계서 작성 가이드 및 시험 수행 교육</li> <li>- 테스트 환경 준비 및 점검</li> <li>- 통합 테스트 수행 /시스템 테스트 수행</li> <li>- 인수 테스트 수행 지원</li> <li>- 테스트 결과 보고</li> </ul>	000, 000	
개발자	<ul style="list-style-type: none"> <li>- 컴포넌트 테스트 수행</li> <li>- 결함 사항 수정 및 확인</li> </ul>	000, 000, 000	타 개발자가 테스트를 수행함.

2. 각 단계별 테스트 조직 및 역할을 정의한다.

(1) 단위 테스트 조직을 구성한다.

응용 시스템 개발 팀에서 SW 준비, 테스트 데이터 준비 등의 테스트 환경을 준비하고 결함을 수정하는 역할을, 프로젝트 테스트 팀에서 단위 테스트를 수행하고 결함 수정 여부를 재검사하는 역할을 한다.

(2) 통합 테스트 조직을 구성한다.

응용 시스템 개발 팀에서 통합 테스트 환경을 준비하는 한편 결함 수정을 하고, 본사에서 지원하는 본사 테스트 팀에서 통합 테스트를 수행할 수 있다.

(3) 시스템 테스트 조직을 구성한다.

프로젝트 테스트 팀에서 시스템 테스트를 계획, 설계, 수행 및 보완 등의 역할을 수행한다.

(4) 인수 테스트 조직을 구성한다.

응용 시스템 개발 팀, 프로젝트 테스트 팀, 고객 등으로 조직을 구성하고, 인수 테스트 계획, 설계, 수행 및 결과 보고의 역할을 수행할 수 있다.

<표 1-7> 단계별 테스트 조직 구성 예시

테스트 단계	역할 정의	테스트 조직
총괄 테스트 계획	<ul style="list-style-type: none"> <li>- 시험 전략 및 수행 절차 수립</li> <li>- 시험 조직 구성</li> <li>- 시험 전체 일정 수립</li> <li>- 총괄 시험 계획서 작성</li> </ul>	테스트 관리자
단위 테스트 계획	<ul style="list-style-type: none"> <li>- 응용 개발 팀별 단위 테스트 케이스 도출</li> <li>- 개발 팀장과 단위 테스트 검토</li> </ul>	테스트 팀
통합 테스트 계획	<ul style="list-style-type: none"> <li>- 통합 테스트 시나리오 작성</li> </ul>	테스트 팀
테스트 교육	<ul style="list-style-type: none"> <li>- 단위 테스트 수행 방법</li> <li>- 업무별 테스트 수행 절차 교육</li> <li>- 테스트 산출물 작성 방법 교육</li> </ul>	테스트 팀
단위 테스트	<ul style="list-style-type: none"> <li>- 시험 환경 준비           <ul style="list-style-type: none"> <li>. 소프트웨어 준비</li> <li>. 시험 데이터 준비</li> </ul> </li> <li>- 단위 테스트 수행</li> </ul>	응용 개발 팀
	<ul style="list-style-type: none"> <li>- 결함 수정</li> </ul>	프로젝트 테스트 팀
	<ul style="list-style-type: none"> <li>- 결함 수정 여부 재검사</li> </ul>	응용시스템 개발 팀
통합 테스트 환경 구축	<ul style="list-style-type: none"> <li>- 운영 장비 설치</li> <li>- 시스템 이관(개발 및 운영)           <ul style="list-style-type: none"> <li>. OS, DBMS 설치 및 테스트</li> <li>. 응용 프로그램 통합 및 환경 구축</li> </ul> </li> <li>- DB 백업</li> </ul>	프로젝트 테스트 팀
통합 테스트	<ul style="list-style-type: none"> <li>- 통합 테스트 수행</li> </ul>	응용시스템 개발 팀
	<ul style="list-style-type: none"> <li>- 결함 수정</li> </ul>	본사 테스트 팀
	<ul style="list-style-type: none"> <li>- 재검사</li> </ul>	프로젝트 테스트 팀
시스템 테스트	<ul style="list-style-type: none"> <li>- 시스템 테스트 계획</li> <li>- 시스템 테스트 설계</li> <li>- 시스템 테스트 수행</li> <li>- 시스템 테스트 결과 보고</li> <li>- 시스템 테스트 보완 요청 사항 보완</li> </ul>	프로젝트 테스트 팀
인수 테스트	<ul style="list-style-type: none"> <li>- 인수 테스트 준비</li> <li>- 인수 테스트 수행</li> <li>- 인수 테스트 결과 보고</li> </ul>	응용시스템 개발 팀, 프로젝트 테스트 팀, 고객

**[3] 통합 테스트를 위한 테스트 데이터, 시작 및 종료 조건을 준비한다.**

1. 테스트 수행을 위한 테스트 데이터를 준비한다.

(1) 기존 시스템에서 운영되는 실제 데이터를 사용하여 준비한다.

현재 운영 중인 시스템 또는 서버의 데이터를 복사하여 준비하고, 필요 시 데이터 프라이버시 규제를 준수하기 위하여 연산을 통해 개인정보를 마스킹 처리한다.

(2) 가상의 데이터를 생성하고 준비한다.

데이터 생성 프로그램 또는 DBMS의 SQL을 이용하여 데이터를 생성하고, 보안을 준수 할 수 있도록 데이터 프로파일링 처리를 한다.

2. 테스트 수행을 위한 시작 조건 및 종료 조건을 준비한다.

(1) 테스트 수행을 위한 시작 조건을 정의한다.

(2) 테스트 완료를 위한 종료 조건을 정의한다.

**[4] 테스트 방식, 대상과 범위를 반영한 테스트 시나리오를 정의한다.**

개발하고자 하는 응용 소프트웨어의 특성을 반영하여 통합 테스트를 위한 테스트 시나리오를 정의한다.

1. 테스트 시나리오 작성 방법을 결정한다.

개발된 애플리케이션의 테스트 대상, 업무 기능, 테스트 방법에 따라 테스트 시나리오를 정의한다.

(1) 각 업무 단위별 테스트 시나리오를 정의한다.

(가) 진행 중인 프로젝트의 개발 수명 주기를 판단한다.

(나) 작업 분할 구조와 현재 프로젝트 진행 상황을 판단한다.

(다) 테스트 일정과 테스트 투입 인력 등 테스트 비용을 산정한다.

(2) 대상 시스템별 테스트 시나리오를 정의한다.

테스트 대상이 되는 시스템을 분류하여 각 시스템별(고객 관리 시스템, 고객센터, 인사 /급여 관리 시스템 등), 기능 분할 모듈별, 테스트 항목별 테스트 시나리오를 정의한다.

(3) 테스트 방법에 따른 테스트 시나리오를 정의한다.

사용자 요구사항에 맞는 기능 테스트, 성능 테스트, 보안 테스트, 품질 테스트, 사용성 테스트 등 적절한 테스트 방법에 따른 테스트 시나리오를 정의한다.

2. 통합 테스트 수행을 위한 테스트 시나리오를 작성한다.

- (1) 구현된 기능의 업무 흐름에 따른 테스트 시나리오를 작성한다.
  - (가) 기본 흐름을 검증할 수 있는 테스트 시나리오를 작성한다.
  - (나) 대체 흐름을 검증할 수 있는 테스트 시나리오를 작성한다.
  - (다) 예외 발생 시 검증 가능한 테스트 시나리오를 작성한다.
- (2) 단순 요구 기능의 구현 여부를 검증하는 테스트 시나리오를 작성한다.
  - (가) 체크리스트 형태의 테스트 시나리오를 작성한다.
  - (나) 화면 정의를 포함하는 형태의 테스트 시나리오를 작성한다.
- (3) 입력 데이터에 따른 기능 검증을 위한 테스트 시나리오를 작성한다.
  - (가) 정상 데이터 범위를 가정한 테스트 시나리오를 작성한다.
  - (나) 비정상 데이터 범위를 가정한 테스트 시나리오를 작성한다.

3. 비기능 테스트 수행을 위한 테스트 시나리오를 작성한다.

비기능 요소를 식별하고, 성능, 보안, 품질, 사용성 요구사항을 검증할 수 있는 테스트 시나리오를 작성한다.

- (1) 성능 요구사항 검증을 위한 테스트 시나리오를 작성한다.
- (2) 품질 요구사항 검증을 위한 테스트 시나리오를 작성한다.
- (3) 보안 요구사항 검증을 위한 테스트 시나리오를 작성한다.
- (4) 사용성 요구사항 검증을 위한 테스트 시나리오를 작성한다.

4. 테스트 시나리오 작성 시 고려할 사항을 정의한다.

- (1) 유스케이스(Use Case)간 업무 흐름이 정상 동작되는지 테스트할 수 있는 시나리오를 정의한다.
- (2) 개발된 각 모듈 또는 프로그램의 연계가 정상적으로 이루어지는지 테스트할 수 있는 시나리오를 정의한다.
- (3) 서브시스템이 존재할 경우 각 시스템의 연계가 정상적으로 이루어지는지 테스트할 수 있는 시나리오를 정의한다.

<표 1-8> 통합 테스트 시나리오 작성 예시

통합 테스트 시나리오				프로젝트 명 : xx시스템 기능 개선 사업									
				시스템 명 : xx시스템									
문서번호 : CS5xx-TS922		작성자 : 흥길동		작성일 : 2016.00.00		버전 : 1.0							
d) 통합 테스트 시나리오													
① 인사													
테스 트 ID	테스 트 명	번 호	메뉴 경로	주요 입력 값	테스트 케이스	예상 결과	확인						
IT_Hxx _01xx	기준 관리	1	인사관리> 기준관리> 부서조직관 리	-조직 개편일: 2015-03 -00	-검색 조건을 입력하고 조회 버튼을 클릭 -엑셀 버튼 클릭	-검색 결과가 조회된다. -검색 결과가 엑셀로 다운로드된다.							
		2	인사관리> 기준관리> 부서변경 이력조회	-조직 개편일: 2015-03 -00	-신규 버튼 -조직 개편일: 2015-03 클릭 -변경 구분, 부서명, 분,부서 명,부서 장	-조직 개편일 입력이 가능해진다. -코드 목록에서 행이 추가된다. -변경 구분, 부서명, 부서장이 저장된다.							
IT_Hxx _01xx	기준 관리	3	인사관리> 기준관리> 통합직급코 드관리	-조직 개편일: 2015-03 -00	-검색 조건을 입력하고 조회 버튼을 클릭 -엑셀 버튼 클릭	-검색 결과가 조회된다. -검색 결과가 엑셀로 다운로드된다.							
		4	인사관리> 기준관리> 부서변경 이력조회	-조직 개편일: 2015-03 -00	-검색 조건을 입력하고 조회 버튼을 클릭 -엑셀 버튼 클릭	-검색 결과가 조회된다. -검색 결과가 엑셀로 다운로드된다.							
IT_Hxx _01xx	기준 관리	5	인사관리> 기준관리> 정원기준관 리	-년도: 20xx -정원:00	-엑셀 버튼 클릭 -행 추가 버튼 클릭 -년도, 정원, 비고를 선택 또는 입력 후 저장 버튼 클릭	-검색 결과가 엑셀로 다운로드된다. -선택된 행 다음 행으로 입력 창이 뜬다. -해당 내용이 저장된다.							

⑤ 작성된 테스트 시나리오를 수행하기 위한 테스트 환경을 준비한다.

1. 테스트 수행을 위한 하드웨어 환경을 준비한다.

(1) 테스트 환경 구현을 위한 서버를 구축한다.

(가) 운영 환경과 동일한 사양의 테스트 서버를 설치한다.

(나) 해당 DBMS 설치 및 사용자 권한을 설정한다.

(2) 기존 시스템을 이관한다.

현재 운영되고 있는 시스템의 일부를 테스트 환경으로 옮겨서 구축한다.

2. 테스트 수행을 위한 소프트웨어 환경을 준비한다.

(1) 테스트 환경 구현을 위한 애플리케이션을 설치한다.

(가) 응용 프로그램 통합 및 환경을 구축한다.

개발 서버에서 완성된 프로그램을 테스트 서버에 설치하고, 사용자 권한 설정 등 테스트 시 WAS(Web Application Server) 또는 DBMS(Database Management System) 서버에 접근 가능한 환경을 구축한다.

(나) 기타 프로그램을 설치한다.

사용자의 테스트 요구사항 중 특별한 내용을 중점적으로 테스트할 수 있는 프로그램 등을 설치한다.

(2) 가상 환경에서의 테스트를 위한 테스트 환경을 구축한다.

(가) 가상 머신 테스트 환경을 구축한다.

기존 개발 서버 또는 운영 서버에 가상 머신 프로그램을 설치하고, 라우터 또는 스위치 등 네트워크 장비에 VLAN을 설정한다.

(나) 클라우드 기반의 테스트를 위한 환경을 구축한다.

클라우드 테스트 환경을 이용하기 위한 계정 설정 및 각종 옵션 기능을 설정한다.

### 수행 tip

- 테스트 시나리오는 구체적으로 어떤 기능 항목을 어떤 순서대로 테스트할 것인지 절차를 기술하고, 사전 조건과 입력 데이터를 미리 설정하도록 한다.
- 테스트 수행을 위한 테스트 데이터는 기존에 사용 중인 데이터 중 일부를 선택한 후 테스트 케이스에 맞게 미리 설정하여 준비하도록 한다.

## 학습1 교수 · 학습 방법

### 교수 방법

- 소프트웨어 테스트의 개념, 테스트의 필요성, 테스트 관련 국제 표준 등 지식 체계에 대해 이해할 수 있도록 충분히 설명한 후 실습을 진행한다.
- 소프트웨어 개발 수명 주기의 단계별 테스트 유형에 대해 설명하고, 학습자의 이해를 바탕으로 실습을 진행한다.
- 학습자의 테스트 케이스에 대한 선수 지식 습득 여부를 확인한 후 수업을 진행한다.
- 테스트 케이스 및 테스트 시나리오의 작성 과정을 시연을 통해 설명하고, 작성할 수 있도록 지도한다.
- 테스트 케이스 작성 시, 테스트 유형별로 케이스를 세분화하여 작성하고, 애플리케이션의 유형별 특성과 해당 특성에 대한 각 테스트 단계별 활동, 테스트 수행의 성공과 실패 기준 등을 정의한 테스트 케이스 명세서를 작성할 수 있도록 지도한다.

### 학습 방법

- 소프트웨어 테스트 핵심 용어를 익히고, 테스트 계획서, 테스트 관련 국제 표준 등의 지식 체계에 대해 숙지한다.
- 소프트웨어 개발 수명 주기 단계별 테스트 방법에 대해 숙지한 후 학습한다.
- 테스트 케이스 작성과 관련한 선수 지식을 학습하고 실습한다.
- 테스트 프로세스 전 과정에 대한 시연을 따라 학습한 후 테스트 케이스 및 테스트 시나리오 작성을 실습한다.
- 테스트 케이스 작성 시, 테스트 유형별로 케이스를 세분화하여 작성하고, 애플리케이션의 유형별 특성과 해당 특성에 대한 각 테스트의 단계별 활동, 테스트 수행의 성공과 실패 기준 등을 정의한 테스트 케이스 명세서를 작성할 수 있도록 실습한다.

## 학습1 평 가

### 평가 준거

- 평가자는 학습자가 학습 목표를 성공적으로 달성하였는지를 평가해야 한다.
- 평가자는 다음 사항을 평가해야 한다.

학습 내용	학습 목표	성취수준		
		상	중	하
애플리케이션 테스트 케이스 작성	- 개발하고자 하는 응용 소프트웨어의 특성을 반영한 테스트 방식, 대상과 범위를 결정하여 테스트 케이스를 작성할 수 있다.			
애플리케이션 테스트 시나리오 작성	- 개발하고자 하는 응용 소프트웨어의 특성을 반영한 테스트 방식, 대상과 범위가 적용된 시나리오를 작성 할 수 있다.			
	- 애플리케이션 테스트 수행에 필요한 테스트 데이터, 테스트 시작 및 종료 조건 등을 준비할 수 있다.			

### 평가 방법

- 포트폴리오

학습 내용	평가 항목	성취수준		
		상	중	하
애플리케이션 테스트 케이스 작성	- 소프트웨어 개발 수명 주기 단계별 테스트 유형에 대한 이해 능력			
	- 통합 테스트 프로세스에 대한 이해 능력			
	- 업무 기능별 단위/통합 테스트 케이스 작성의 완성도			
애플리케이션 테스트 시나리오 작성	- 업무 기능 유형별 테스트 시나리오 작성 능력			
	- 유형별 테스트 환경 구축에 대한 이해 능력			
	- 유형별 테스트 데이터 작성 능력			

• 평가자 체크리스트

학습 내용	평가 항목	성취수준		
		상	중	하
애플리케이션 테스트 케이스 작성	- 단계별 테스트 유형에 대한 이해 능력			
	- 통합 테스트 프로세스에 대한 이해 능력			
	- 업무 기능별 단위/통합 테스트 케이스 작성 능력			
애플리케이션 테스트 시나리오 작성	- 유형별/업무 기능별 테스트 시나리오 작성 능력			
	- 유형별 테스트 데이터 작성에 대한 이해 능력			

## 피드백

### 1. 포트폴리오

- 작성된 테스트 케이스 및 테스트 시나리오를 평가하여 일정 수준 이하인 학습자의 결과물에 누락되거나 잘못 작성된 부분 및 주요 개선 사항에 대해 표시하여 돌려주고 추가 학습 후 재작성하도록 한다.

### 2. 평가자 체크리스트

- 테스트 케이스, 테스트 시나리오 및 테스트 데이터 작성 과정에서 각 단계별 테스트 유형에 대한 이해 능력과 테스트 산출물 작성 능력을 평가하고, 부족한 부분에 대해 피드백을 제공한다.

학습 1	애플리케이션 테스트 케이스 설계하기
학습 2	<b>애플리케이션 통합 테스트하기</b>
학습 3	애플리케이션 성능 개선하기

## 2-1. 애플리케이션 통합 테스트 수행

### 학습 목표

- 개발자 통합 테스트 계획에 따라 통합 모듈 및 인터페이스가 요구사항을 충족하는지에 대한 테스트를 수행할 수 있다.

### 필요 지식 /

#### ① 통합 테스트 수행 방법

##### 1. 개요

###### (1) 통합 테스트의 개념

애플리케이션 통합 테스트는 소프트웨어 각 모듈 간의 인터페이스 관련 오류 및 결함을 찾아내기 위한 체계적인 테스트 기법이다.

###### (2) 통합 테스트의 목적

단위 테스트가 끝난 모듈 또는 컴포넌트 단위의 프로그램이 설계 단계에서 제시한 애플리케이션과 동일한 구조와 기능으로 구현된 것인지를 확인하는 것이다.

###### (3) 통합 테스트 수행 방법의 분류

일반적으로 점증적인 방법과 비점증적인 방식으로 나눌 수 있다. 비점증적인 빅뱅 방식은 모든 컴포넌트를 사전에 통합하여 전체 프로그램을 한꺼번에 테스트하는 것을 말하며, 점증적인 방법은 다시 상향식 통합과 하향식 통합 방식으로 구분할 수 있다.

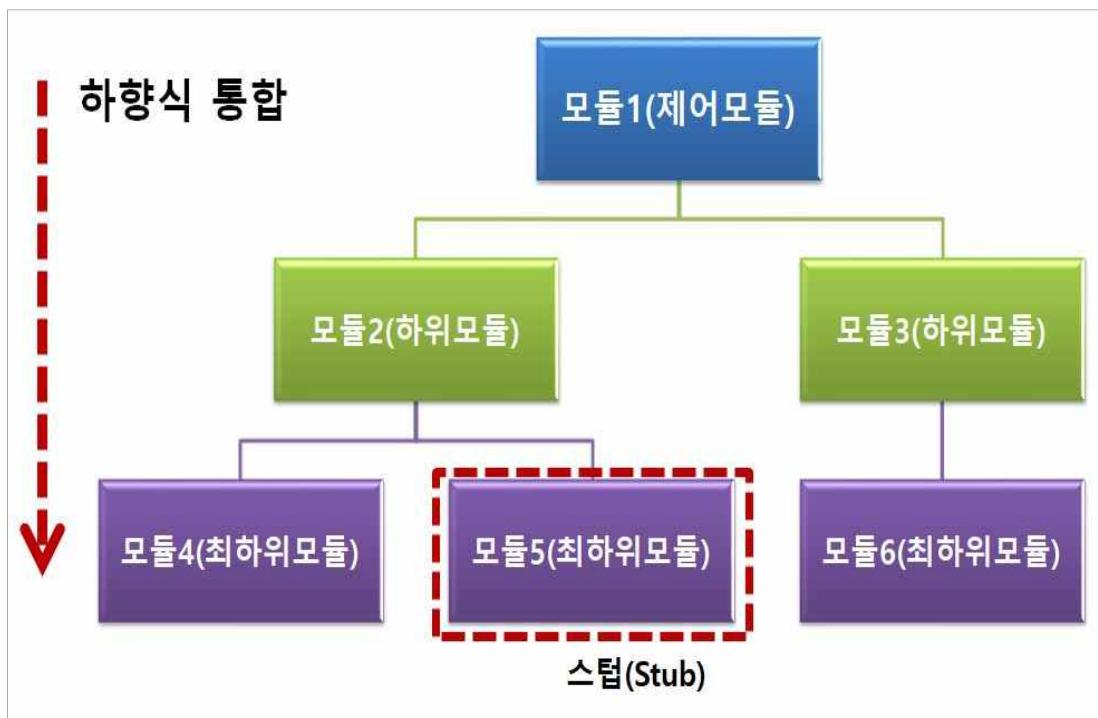
##### 2. 하향식 통합(Top Down)

###### (1) 방식

메인 제어 모듈(프로그램)로부터 아래 방향으로 제어의 경로를 따라 이동하면서 하향식으로 통합하면서 테스트를 진행하며, 메인 제어 모듈에 통합되는 하위 모듈과 최하위 모듈은 ‘깊이-우선’ 또는 ‘너비-우선’ 방식으로 통합된다.

## (2) 수행 단계

- (가) 메인 제어 모듈은 작성된 프로그램을 사용하고, 아직 작성되지 않은 하위 제어 모듈 및 모든 하위 컴포넌트를 대신하여 더미 모듈인 스텁(Stub)을 개발한다.
- (나) 깊이-우선 방식 또는 너비-우선 방식에 따라, 하위 모듈인 스텁이 한 번에 하나씩 실제 모듈로 대체된다.
- (다) 각 모듈 또는 컴포넌트를 통합하면서 테스트가 수행된다.
- (라) 테스트가 완료되면 스텁이 실제 모듈 또는 컴포넌트로 작성된다.



[그림 2-1] 하향식 방식의 통합 테스트

## 3. 상향식 통합(Bottom Up)

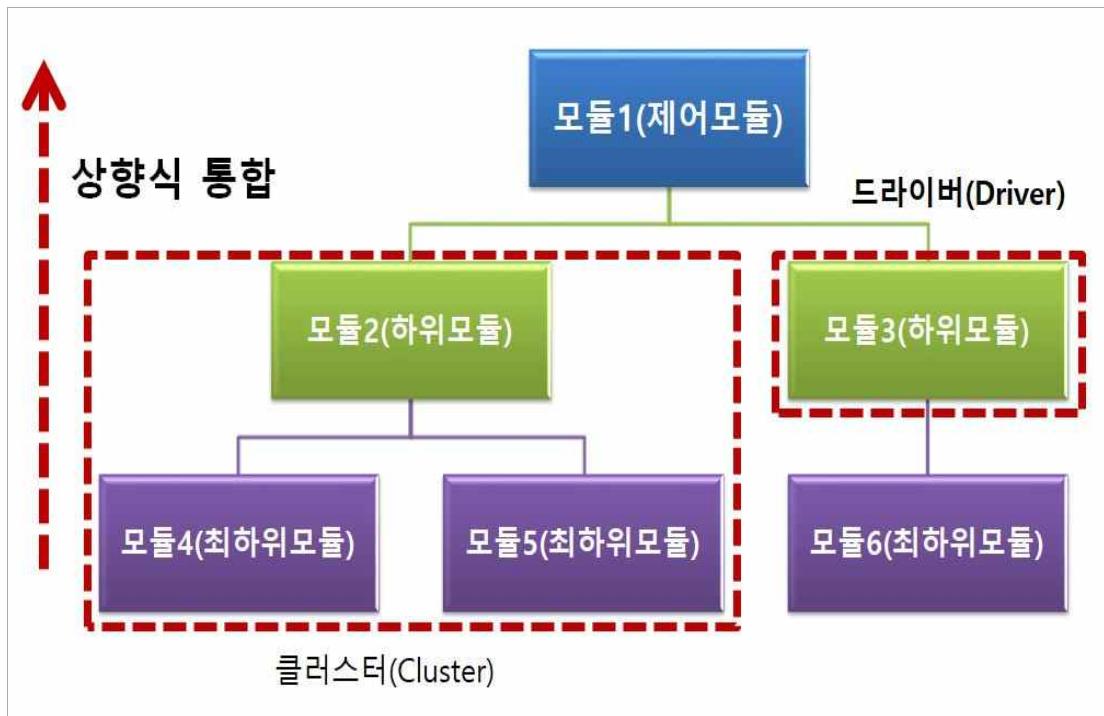
### (1) 방식

애플리케이션 구조에서 최하위 레벨의 모듈 또는 컴포넌트로부터 위쪽 방향으로 제어의 경로를 따라 이동하면서 구축과 테스트를 시작한다.

### (2) 수행 단계

- (가) 최하위 레벨의 모듈 또는 컴포넌트들이 하위 모듈의 기능을 수행하는 클러스터(Cluster)로 결합된다.
- (나) 상위의 모듈에서 데이터의 입력과 출력을 확인하기 위한 더미 모듈인 드라이버(Driver)를 작성한다.
- (다) 각 통합된 클러스터 단위를 테스트한다.

(라) 테스트가 완료되면 각 클러스터들은 프로그램의 위쪽으로 결합되며, 드라이버는 실제 모듈 또는 컴포넌트로 대체된다.



[그림 2-2] 상향식 방식의 통합 테스트

#### 4. 회귀 테스팅(Regression Testing)

통합 테스트가 완료된 후에 변경된 모듈이나 컴포넌트가 있다면 새로운 오류 여부를 확인하기 위해 회귀 테스트를 수행할 수 있다.

##### (1) 정의

- (가) 통합 테스트 과정에서 오류를 제거하거나 수정한 프로그램이 새로운 형태의 오동작이나 오류를 일으킬 수 있다.
- (나) 회귀 테스트는 모듈이나 컴포넌트의 변화로 인해 의도하지 않은 오류가 생기지 않았음을 보증하기 위해 반복 테스트하는 것을 말한다.

##### (2) 회귀 테스트 케이스 선정 방법

- (가) 모든 애플리케이션의 기능을 수행할 테스트 케이스의 대표적인 샘플을 도출한다.
- (나) 변경에 의한 영향도가 가장 높은 애플리케이션 기능에 집중한 추가적인 테스트 케이스를 도출한다.
- (다) 실제 수정이 발생한 모듈 또는 컴포넌트에서부터 시행하는 테스트 케이스를 도출한다.

## ② 테스트 자동화 도구

### 1. 테스트 자동화

#### (1) 테스트 자동화의 개념

테스트 자동화란 테스트 도구를 활용하여 반복적인 테스트 작업을 스크립트 형태로 구현함으로써, 테스트 시간 단축과 인력 투입 비용을 최소화하는 한편, 쉽고 효율적인 테스트를 수행할 수 있는 방법이다.

#### (2) 테스트 도구의 장점

- (가) 반복되는 테스트 데이터 재입력 작업의 자동화
- (나) 사용자 요구 기능의 일관성 검증에 유리
- (다) 테스트 결과 값에 대한 객관적인 평가 기준 제공
- (라) 테스트 결과의 통계 작업과 그래프 등 다양한 표시 형태 제공
- (마) UI가 없는 서비스의 경우에도 정밀한 테스트 가능

#### (3) 테스트 도구의 단점

- (가) 도구 도입 후 도구 사용 방법에 대한 교육 및 학습이 필요
- (나) 도구를 프로세스 단계별로 적용하기 위한 시간, 비용, 노력이 필요
- (다) 상용 도구의 경우 고가, 유지 관리 비용이 높아 추가 투자가 필요

### 2. 테스트 자동화 도구 유형

#### (1) 정적 분석 도구(Static Analysis Tools)

- (가) 정적 분석 도구는 만들어진 애플리케이션을 실행하지 않고 분석하는 방법이다.
- (나) 대부분의 경우 소스 코드에 대한 코딩 표준, 코딩 스타일, 코드 복잡도 및 남은 결함을 발견하기 위하여 사용한다.
- (다) 테스트를 수행하는 사람이 작성된 소스 코드에 대한 이해를 바탕으로 도구를 이용해서 분석하는 것을 말한다.

#### (2) 테스트 실행 도구(Test Execution Tools)

이 도구는 테스트를 위해 작성된 스크립트를 실행한다. 작성된 스크립트는 각 스크립트마다 특정 데이터와 테스트 수행 방법을 포함하고 있으며, 데이터 주도 접근 방식과 키워드 주도 접근 방식으로 나눌 수 있다.

##### (가) 데이터 주도 접근 방식

테스트 데이터를 스프레드시트에 저장하고, 이 데이터를 읽고 실행할 수 있도록 한다. 다양한 테스트 데이터를 이용하여 동일한 테스트 케이스를 반복해서 실행할 수 있으며, 스크립트 언어에 익숙지 않은 테스터도 미리 작성된 스크립트에 테스트 데

이터만 추가하여 쉽게 테스트를 수행할 수 있게 된다.

#### (나) 키워드 주도 접근 방식

일반적으로 테스트를 수행할 동작을 나타내는 키워드와 테스트 데이터를 스프레드 시트에 저장한다. 이 방식에서는 키워드를 이용하여 테스트 수행 동작을 정의할 수 있으며, 테스트 대상 애플리케이션의 특성에 맞추어 키워드에 대해 테일러링 (Tailoring)을 수행할 수 있다.

### (3) 성능 테스트 도구(Performance Test Tools)

성능 테스트 도구는 애플리케이션의 처리량, 응답 시간, 경과 시간, 자원 사용률에 대해 가상의 사용자를 생성하고 테스트를 수행함으로써 성능 목표를 달성하였는지를 확인하는 도구이다. 일반적으로 시스템 테스트 단계에 성능 테스트 계획을 설계하고 실행하고 있으며, 테스트 결과를 해석할 수 있는 성능 전문가의 도움이 필요한 경우도 있다.

### (4) 테스트 통제 도구(Test Control Tools)

테스트 통제 도구에는 테스트 계획 및 관리를 위한 테스트 관리 도구, 테스트 수행에 필요한 데이터와 도구를 관리하는 형상 관리 도구, 테스트에서 발생한 결함에 대해 관리하거나 협업을 지원하기 위한 결합 추적/관리 도구 등이 있다. 또한 조직의 요구사항에 최적화된 형태의 정보를 생성, 관리하기 위하여 스프레드시트 등 다른 도구들과 연계하여 사용할 수도 있다.

### (5) 테스트 장치(Test Harness)

#### (가) 정의

애플리케이션 컴포넌트 및 모듈을 테스트하는 환경의 일부분으로, 테스트를 지원하기 위한 코드와 데이터를 말하며, 단위 또는 모듈 테스트에 사용하기 위해 코드 개발자가 작성한다.

#### (나) 구성요소

##### 1) 테스트 드라이버(Test Driver)

테스트 대상 하위 모듈을 호출하고, 파라미터를 전달하고, 모듈 테스트 수행 후의 결과를 도출하는 등 상향식 테스트에 필요하다.

##### 2) 테스트 스탍(Test Stub)

제어 모듈이 호출하는 타 모듈의 기능을 단순히 수행하는 도구로 하향식 테스트에 필요하다.

##### 3) 테스트 슈트(Test Suites)

테스트 대상 컴포넌트나 모듈, 시스템에 사용되는 테스트 케이스의 집합을 말한다.

##### 4) 테스트 케이스(Test Case)

입력 값, 실행 조건, 기대 결과 등의 집합을 말한다.

### 5) 테스트 스크립트(Test Script)

자동화된 테스트 실행 절차에 대한 명세를 말한다.

### 6) 목 오브젝트(Mock Object)

사용자의 행위를 조건부로 사전에 입력해 두면, 그 상황에 예정된 행위를 수행하는 객체를 말한다.

## (6) 테스트 자동화 도구

테스트 자동화 도구는 휴먼 에러(Human Error)를 줄이고, 테스트에 소요되는 비용과 시간을 절감하며, 테스트 품질을 향상할 수 있는 도구이다.

테스트 계획, 테스트 분석/설계, 테스트 수행, 테스트 통제 등의 테스트 활동에 따라 각각 요구사항 관리, 테스트 케이스 생성, 커버리지 분석, 정적/동적 테스트 수행, 성능 테스트, 모니터링, 형상 관리, 테스트 관리, 결합 추적/관리 등을 지원하는 테스트 도구가 있다.

<표 2-1> 테스트 단계별 테스트자동화 도구

테스트 단계	자동화 도구	도 구 설 명
테스트 계획	요구사항 관리	고객 요구사항 정의 및 요구사항 관리를 지원
테스트 분석/설계	테스트케이스 생성	테스트 기법에 따른 테스트 케이스 작성과 테스트 데이터 생성을 지원
	테스트 자동화	기능 테스트와 UI 테스트 등 단위 테스트 및 통합테스트를 지원
	정적 분석	코딩표준, 런타임 오류 등을 검증
테스트 수행	동적 분석	대상시스템 시뮬레이션을 통한 오류 검출
	성능 테스트	부하생성기 등을 이용하여 가상 사용자를 생성하고, 시스템의 처리능력을 측정하는 도구
	모니터링	시스템 자원(CPU, Memory 등)의 상태 확인 및 분석 지원
	커버리지 측정	테스트 완료 후 테스트 충분성 여부 검증 지원
테스트 관리	형상관리	테스트 수행에 필요한 도구, 데이터 및 문서 관리
	결합 추적/관리	테스트에서 발생한 결합 추적 및 관리 활동 지원

## 수행 내용 / 애플리케이션 통합 테스트 수행하기

### 재료 · 자료

- SW 개발 국제 표준(ISO/IEC 9126, ISO/IEC 14598, ISO15504, ISO/IEC 12207, CMMI 등)
- 소프트웨어 테스트 관련 국제 표준(ISO/IEC 29119)
- 요구사항 정의서, 통합 테스트 계획서, 테스트 케이스, 테스트 시나리오, 테스트 데이터

### 기기(장비 · 공구)

- 전산 장비(컴퓨터, 프린터, 빔 프로젝터, 네트워크 장비 등)
- OA 도구(워드프로세서, 스프레드시트, 프레젠테이션 도구 등)
- CASE 도구(SW 테스트 지원 도구, SW설계 지원 도구, SW 개발 지원 도구 등)

### 안전 · 유의 사항

- 통합 테스트 수행은 정의된 방법과 절차에 따라 실시하여야 한다.
- 테스트 지원 도구의 사용 방법과 유의사항에 대해 습득하여야 한다.
- 테스트 방법별 통합 테스트 시나리오를 작성하여야 한다.
- 테스트 수행은 개발 및 실제 운영 환경과는 분리된 시스템에서 수행하여야 한다.
- 테스트를 실행하는 서버는 운영 서버와 동일 또는 유사한 성능을 보유한 사양으로 준비하여야 한다.
- 테스트 수행을 위한 데이터는 미리 준비하고 검증하여야 한다.

### 수행 순서

#### ① 통합 테스트 계획서를 검토한다.

통합 테스트에 투입될 인력 및 테스트 일정, 테스트 수행 방안, 테스트 절차, 테스트 환경, 테스트 판정 기준, 테스트 완료 조건에 대해 검토한 후, 실제 테스트를 수행할 인력, 일정, 테스트 방법에 대해 결정한다.

#### 1. 통합 테스트에 투입될 일정 및 인력을 확정한다.

수행 중인 프로젝트 일정 및 통합 테스트 계획에 따라 테스트 인력을 확정한다.

(1) 통합 테스트 수행 일정을 확정한다.

프로그램 배포 또는 진행 중인 프로젝트의 일정 계획에 따라 테스트 수행 기간 및 테스트 회차 등 통합 테스트 전체 일정을 확정한다.

(2) 테스트 수행 조직을 구성하고 담당자별 업무 분장을 확정한다.

담당자별 통합 테스트에서 수행할 역할과 책임에 대해 결정한다.

(‘〈표 1-6〉 테스트 수행을 위한 업무 분장 작성 예시’ 참조)

2. 통합 테스트 수행 방안에 대해 검토 후 테스트 방법을 결정한다.

(1) 통합 테스트 수행 방안에 대해 검토한다.

프로젝트 특성에 맞게 각 테스트 방식(상향식, 하향식, 빅뱅 방식)에 대한 장점과 단점을 비교하여 적절한 방법을 결정한다.

(가) 상향식 테스트 방안의 장점과 단점에 대해 정리한다.

(나) 하향식 테스트 방안의 장점과 단점에 대해 정리한다.

(다) 빅뱅 테스트 방안의 장점과 단점에 대해 정리한다.

(2) 프로젝트의 특성에 맞는 테스트 수행 방법을 결정한다.

각 단위 기능의 연계 동작이 적정한지를 검토할 테스트 방법을 결정한다.

<표 2-2> 통합 테스트 수행 방법 분류

테스트 방안	빅뱅(Big Bang)	상향식(Bottom Up)	하향식(Top Down)
테스트 수행 방법	모든 모듈을 동시에 통합 후 테스트 수행	최하위 모듈부터 점진적으로 상위 모듈과 함께 테스트	최상위 모듈부터 하위 모듈들을 통합하면서 테스트
드라이버 /스텝	드라이버/스텝 없이 실제 모듈로 테스트	테스트 드라이버 필요	테스트 스텝 필요
장점	단시간 테스트 가능, 작은 시스템에 유리	장애 위치 파악 쉬움, 모든 모듈 개발 시간 낭비 필요 없음.	장애 위치 파악 쉬움, 이른 프로토타입 가능, 중요 모듈의 선 테스트 가능, 설계상 결함 조기 발견
단점	장애 위치 파악이 어려움, 모든 모듈이 개발되어야 가능	중요 모듈들이 마지막 테스트될 가능, 이른 프로토타입 어려움.	많은 스텝이 필요, 하위 모듈들의 불충분한 테스트 수행

② 통합 테스트를 수행할 테스트 환경을 준비한다.

운영 환경과 동일한 환경에서 테스트를 수행하는 것이 가장 바람직하지만 비용, 일정 등 다양한 프로젝트 진행 상황에 따라 개발 환경에서 진행할 수도 있다.

1. 테스트 서버를 설치한다.

테스트를 진행하는 서버는 별도의 테스트 서버를 설치하여 진행하거나 프로젝트 진행 중인 개발 서버를 대상으로 구축할 수도 있다.

(1) 운영 체제 서버(OS Server)를 설치한다.

Windows 서버 또는 Linux 서버 또는 기타 운영 체제를 설치한다.

(2) 웹 서버(Web Server)를 설치한다.

IIS(Internet Information Server), Apache, Webtobe 등 운영 환경과 동일한 서버를 설치한다.

(3) 웹 애플리케이션 서버(WAS : Web Application Server)를 설치한다.

Tomcat, Jeus, Web Logic, Web Spere 등 운영 환경과 동일한 서버를 설치한다.

(4) DBMS(DataBase Management System)를 설치한다.

Oracle, MS-Sql, MySql, DB2, Informix, Sybase, Derby, SQLite 등 운영 환경과 동일하게 설치한다.



[그림 2-3] 별도의 테스트 서버 설치

## 2. 설치한 서버의 시스템 환경을 설정한다.

테스트를 진행할 서버의 시스템 정보, 데이터베이스 정보, 로그인 확인 방법, 테스트 계정 등의 환경을 설정한다.

### (1) 시스템 정보를 설정한다.

IP주소, 접속 방법, WAS 서버의 위치, 첨부파일의 위치, 테스트 URL 등의 정보를 설정 한다.

### (2) 데이터베이스 연결 정보를 설정한다.

IP주소, port 번호, 데이터베이스 연결 계정 등 정보를 설정한다.

### (3) 각종 로그 확인 방법에 대해 설정한다.

Application Log, Error Log, DataBase Sql Log의 확인 방법을 설정한다.

### (4) 테스트 담당자가 사용할 계정을 설정한다.

일반 사용자 테스트용 계정과 관리자용 테스트 계정을 설정한다.

<표 2-3> 테스트 서버 환경 설정 예시

## 6. 테스트 환경

테스트를 진행하는 서버는 개발 서버를 대상으로 함.

	OS	webserver/WAS	DBMS
	Linux Redhat	Webtobe / Jeus 6.x	Oracle 11G

---

6.1 테스트 환경 상세		
항목	구분	상세 설명
시스템 정보	IP	172.16x.xxx.xxx
	접속 방법	Sxxx, Sxx
	제우스 위치	/txxxx/jeusx
	WEBTOBE 위치	/txxxx/webtob
	첨부파일 위치	/xxxch
데이터베이스 연결 정보	URL	http://172.16x.xxx.xxx
	IP	172.16x.xxx.xxx
	port	1xxx
로그 확인 방법	SID	wixxx
	application	/txxxx/jeusx/logs/
	spy	/txxxx/jeusx/logs/
테스트 계정	error	/txxxx/jeus/logs
	관리자	00xx / 1 (정00)
	일반 사용자	50xx / 1 (강00)

**[3] 통합 테스트 케이스 및 시나리오를 검토한다.**

1. 통합 테스트 케이스를 검토한다.

프로젝트에서 구현해야 하는 기능 요구사항에 대한 올바른 설계 여부를 통합 테스트를 통해 확인 가능한지 검토한다.

(1) 테스트 ID와 테스트 사양서, 테스트 명, 테스트 내용의 일관성을 검토한다.

테스트 ID별 통합 테스트 명세서 또는 테스트 사양서의 내용과 테스트할 내용이 해당 기능의 정상적인 동작 여부를 확인할 수 있도록 작성되어 있는지 검토한다.

(2) 작성된 테스트 케이스에 대해 동료 검토를 통해 테스트 유효성, 효율성 등을 확인 한다.

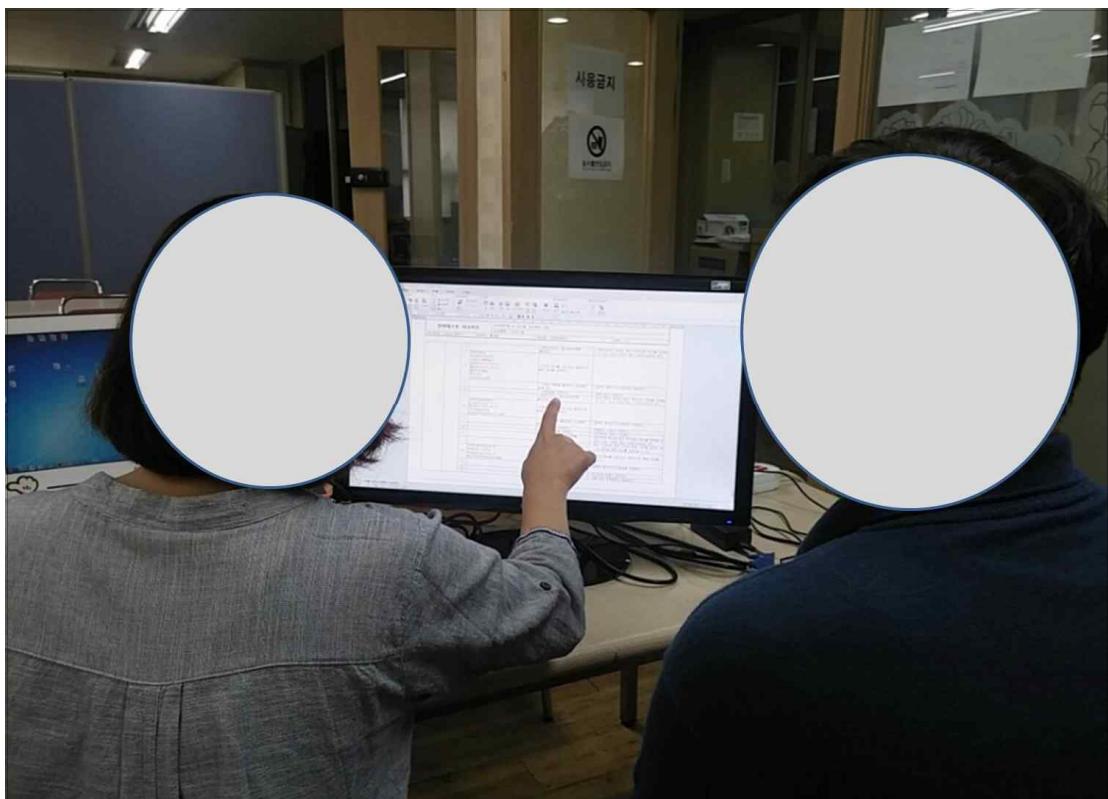
2. 통합 테스트 시나리오를 검토한다.

메뉴 기반이 통합 테스트의 경우, 수행할 통합 테스트의 메뉴 경로 및 주요 입력 값 등 해당 테스트 시나리오의 유효성에 대해 확인한다.

(1) 통합 테스트 수행을 위한 권한 정의가 적정한지 검토한다.

(2) 테스트 수행을 위한 사이트 접속 정보 등이 적정한지 검토한다.

(3) 작성된 테스트 시나리오에 대해 동료 검토를 통해 확인한다.



[그림 2-4] 통합 테스트 시나리오에 대한 동료 검토

④ 통합 모듈 및 인터페이스가 요구사항을 충족하는지 테스트를 수행한다.

1. 통합 테스트 계획서에 명시된 일정에 따라 테스트를 수행한다.

검토 완료된 통합 테스트 케이스 및 테스트 시나리오에 따라 테스트를 수행한다.

2. 메뉴 기반 테스트인 경우 메뉴의 정상 흐름 또는 비정상적인 흐름에 대해 테스트를 수행한다.

테스트 시나리오를 기반으로 작성된 메뉴 흐름에 따라 정상적인 데이터 또는 비정상적인 데이터를 이용하여 테스트를 수행한다.

3. 경험 기반 테스트인 경우 별도의 테스트 케이스 없이 테스트 담당자가 유사 프로젝트의 경험을 토대로 테스트를 수행한다.

경험 기반 테스트 중 탐색적 테스트인 경우, 아래의 구성요소에 대해 작성한다.

(1) 테스트 차터를 작성한다.

(2) 테스트 제한 시간을 결정한다.

(3) 테스트 노트를 작성한다.

(4) 테스트 요약 보고를 작성한다.

4. 테스트 자동화 도구를 사용하여 통합 테스트를 수행한다.

각 채널 간 인터페이스의 결합을 통해 기능을 검증하는 경우에는 인터페이스 테스트 자동화 도구를 사용하여 통합 테스트를 수행한다.

(1) 통합 테스트 환경(서버 내부 로컬 호출, 서버 리모트 호출, 서버 HTTP 호출에 대해 정의한다.

(2) 테스트를 수행할 입력 정보 및 결과 정보를 이용하여 테스트 테이블을 작성한다.

(3) 테스트 케이스를 작성하고, 테스트를 수행한다.

⑤ 통합 테스트 결과를 기록한다.

통합 테스트 수행 후 테스트 ID별로 수행한 통합 테스트 결과에 대해 작성한다.

1. 테스트 결과서에 공통으로 들어갈 항목에 대해 작성한다.

테스트 ID, 테스트 명, 작성자, 수정자, 작성 일자, 수정 일자 등의 내용을 작성한다.

2. 테스트 결과 판정을 위한 내용을 작성한다.

요구사항 ID, 테스트 내용, 전제 조건, 순번, 테스트 시나리오, 테스트 데이터, 개발자명, 시험자명, 시험 일자, 테스트 예상 결과, 적합 여부, 심각도, 결함 내용, 수정 예정일, 수정 완료일 등의 내용을 작성한다.

<표 2-4> 통합 테스트 결과서 작성 예시

테스트 ID	IT_SIS_xxx	작성자	김xx	작성 일자	2016-05-30						
테스트 명	xxx정보	수정자		수정 일자							
요구사항 ID	REQ_FUC_0xx										
테스트 내용	물질 정보를 검색 조건에 따라 조회한다.										
전체 조건	관리자 권한으로 로그인한다.(관리자id : xxxadmin)										
No	테스트 시나리오	테스트 Data	개발자	시험자	시험 일자	test 예상 결과	적합여부	심각도	결함내용	수정예정	완료일자
01	통계>물질정보> 물질정보원부 클릭		김x x	정x x	06.05. 30	정보목록 화면이 보여진다.	적합				
02	검색 조건 입력 후 검색 버튼 클릭	년도:2015 구분:xx 업체: 대분류:xx 중분류:xx 소분류:xx 세분류:xx		김x x	정x x	06.05. 30	xxx코리아, xxx지 등 12xx건의 목록이 조회된다.	적합			
03	검색 조건 입력 후 검색 버튼 클릭	년도:2015 구분:xx 지역:xx 업체:(주)x 대분류:xx 중분류:xx 소분류:xx 세분류:xx		김x x	정x x	06.05. 30	(주)x 업체에 대한 1건의 목록이 조회된다.	부적합	단순결함	검색안됨	16. 06. 01
04	엑셀 버튼 클릭		김x x	정x x	06.05. 30	(주)x 업체에 대해 조회된 건의 정보가 엑셀로 다운된다.	적합				
05	검색 조건 입력 후 검색 버튼 클릭	업체:xx	김x x	정x x	06.05. 30	xx의 모든 건의 정보가 조회된다.	부적합				
06	검색 조건 입력 후 엔터	업체:xx	김x x	정x x	06.05. 30	xx업체 목록이 팝업된다.	적합				
07	조회된 목록 더블 클릭	xx데이터 선택	김x x	정x x	06.05. 30	xx업체에 대한 82건의 상세정보가 조회된다.	적합				
08	엑셀 다운로드 버튼 클릭		김x x	정x x	06.05. 30	xx업체의 정보가 다운로드 된다.	부적합	단순결함	다운안됨	16. 06. 01	

### 수행 tip

- 테스트 케이스 및 시나리오를 작성한 후 적정하게 작성되었는지를 전문 검토자 또는 동료 검토를 통해 검증 작업을 수행하는 것이 바람직하다.
- 통합 테스트 수행은 업무 기능별로 다른 테스트 방법을 적용하지만, 일반적으로는 빅뱅 방식을 많이 사용한다.

## 2-2. 애플리케이션 테스트 결과 분석

### 학습 목표

- 개발자 통합 테스트 수행 결과 발견된 결함에 대한 추이 분석을 통하여 잔존 결함을 추정할 수 있다.

### 필요 지식 /

#### ① 테스트 결과 분석

##### 1. 소프트웨어 결함

소프트웨어의 결함을 말할 때 에러(Error), 결함(Defect), 결점(Fault), 버그(Bug), 실패(Failure)와 같은 용어가 사용되며, 이런 용어들의 정의를 다음과 같이 정리할 수 있다.

###### (1) 에러(Error)/오류

에러는 결함(Defect)의 원인이 되는 것으로, 일반적으로 사람(소프트웨어 개발자, 분석가 등)에 의해 생성된 실수를 말한다.

###### (2) 결함/결점/버그(Bug)

에러 또는 오류가 원인이 되어 소프트웨어 제품에 포함되어 있는 결함을 말하며, 이를 제거하지 않으면 소프트웨어 제품이 실패(Failure)하거나 문제(Problem)가 발생할 수 있다.

###### (3) 실패/문제

소프트웨어 제품에 포함된 결함이 실행될 때 발생되는 현상을 말한다.

##### 2. 테스트 완료 조건

단위 테스트, 통합 테스트, 시스템 테스트, 인수 테스트 등 각 단계별 테스트를 언제 어떤 상황에서 종료할 것인지를 결정하는 것이다. 이러한 완료 조건은 프로젝트 특성에 따라 일정, 비용, 조직 등에 제약이 있으므로, 최적의 완료 조건을 계획하여야 한다.

##### 3. 테스트 리포팅

###### (1) 테스트 결과 정리

모든 테스트가 완료되면, 테스트 계획과 테스트 케이스 설계부터 단계별 테스트 시나리오, 테스트 결과까지 모두 포함된 문서를 작성한다.

###### (2) 테스트 요약 문서

테스트 계획, 소요 비용, 테스트 결과로 판단 가능한 대상 소프트웨어의 품질 상태를 포함한 문서를 작성한다.

### (3) 품질 상태

품질 상태는 정량화된 품질 지표인 테스트 성공률, 테스트 커버리지, 발생한 결함의 수와 결함의 중요도 등이 포함된다.

### (4) 테스트 결과서

테스트 결과서는 결함과 관련한 사항을 중점적으로 기록하며, 결함의 내용 및 자원 정보(CPU, Memory, Network 사용률과 로그 정보 등), 결함의 재현 순서를 상세하게 기록한다.

### (5) 테스트 실행 절차 및 평가

단계별 테스트 종료 시 테스트 실행 절차를 리뷰하고 결과에 대한 평가를 수행하고, 그 결과에 따라 실행 절차를 최적화하여 다음 테스트에 적용한다.

## ② 결합 관리

### 1. 테스트 결합 관리

#### (1) 개요

테스트 결합 관리란 각 단계별 테스트 수행 후 발생한 결함의 재발 방지를 위해, 유사 결함 발견 시 처리 시간 단축을 위해 결함을 추적하고 관리하는 활동이다.

#### (2) 결합 추적 관리 활동

단위 테스트, 통합 테스트, 시스템 테스트, 운영 테스트의 단계별 착수 기준과 입력물, 그리고 종료 조건 및 산출물에 대해 다음과 같이 정리할 수 있다.

<표 2-5> 결합 추적 관리 활동

테스트 단계	착수 기준	입력물	종료 조건 / 산출물
단위 테스트	- 해당 사항 없음.	- 테스트 활동 로그 - 결함 관리 대장	- 종료 조건 : 심각도 상위 수준의 결함 해결 여부 확인
통합 테스트	- 설계 문서 결함 발견 - 통합 테스트 평가 완료	- 테스트 활동 로그 - 결함 관리 대장	- 산출물 : 결합 관리 대장, 테스트 활동 로그, 테스트 결과 보고서
시스템 테스트	- 요구사항 명세서 결함 발견 - 시스템 테스트 평가 완료	- 테스트 활동 로그 - 결함 관리 대장	
운영 테스트	- 요구사항 명세서 결함 발견 - 운영 테스트 평가 완료	- 테스트 활동 로그 - 결함 관리 대장	

## 2. 결함 관리 프로세스

### (1) 에러 발견

요구사항 분석, 설계, 테스트 실행 중 에러가 발견될 경우, 테스트 전문가와 프로젝트 팀과 논의한다.

### (2) 에러 등록

결함 관리 대장에 발견된 에러를 등록한다.

### (3) 에러 분석

등록된 에러가 단순 에러인지 아니면 실제 결함인지 분석한다.

### (4) 결함 확정

등록된 에러가 실제 결함으로 확정될 경우, 결함 확정 상태로 설정한다.

### (5) 결함 할당

결함을 해결할 담당자를 지정하여 결함을 할당하고 결함 할당 상태로 설정한다.

### (6) 결함 조치

결함에 대해 수정 활동을 수행하고, 수정이 완료된 경우 결함 조치 상태로 설정한다.

### (7) 결함 조치 검토 및 승인

수정이 완료된 결함에 대해 확인 테스트를 수행하고, 정상적으로 결함 조치가 완료된 경우 결함 조치 완료 상태로 설정한다.

## ③ 결함 추이 분석

### 1. 결함 추이 분석

#### (1) 개요

테스트 완료 후 발견된 결함의 결함 관리 측정 지표의 속성 값을 분석하고, 향후 애플리케이션의 어떤 모듈 또는 컴포넌트에서 결함이 발생할지를 추정하는 작업이다.

#### (2) 분석 유형

결함 추이 분석에는 결함의 분포 분석, 결함 추세 분석, 결함 에이징 분석 등의 유형이 있다.

### 2. 결함 관리 측정 지표

#### (1) 결함 분포

각 애플리케이션 모듈 또는 컴포넌트의 특정 속성에 해당하는 결함의 수를 측정하여 결함의 분포를 분석할 수 있다.

#### (2) 결함 추세

테스트 진행 시간의 흐름에 따른 결함의 수를 측정하여 결함 추세를 분석할 수 있다.

#### (3) 결함 에이징

등록된 결함에 대해 특정한 결함 상태의 지속 시간을 측정하여 분석할 수 있다.

## 수행 내용 / 애플리케이션 테스트 결과 분석하기

---

### 재료 · 자료

- SW 개발 국제 표준(ISO/IEC 9126, ISO/IEC 14598, ISO15504, ISO/IEC 12207, CMMI 등)
- 소프트웨어 테스트 관련 국제 표준(ISO/IEC 29119)
- 요구사항 명세서, 통합 테스트 계획서, 테스트 결과서, 결함 관리 대장

### 기기(장비 · 공구)

- 전산 장비(컴퓨터, 프린터, 빔 프로젝터, 네트워크 장비 등)
- OA 도구(워드프로세서, 스프레드시트, 프레젠테이션 도구 등)
- CASE 도구(SW설계 지원 도구, SW 개발 지원 도구, SW 테스트 지원 도구 등)

### 안전 · 유의 사항

- 테스트 분석을 위한 도구의 사용 방법에 대해 습득하여야 한다.
- 테스트 방법별 통합 테스트 계획을 검토하고 학습하여야 한다.
- 테스트 수행은 개발 환경 및 실제 운영 환경과는 분리된 시스템으로 구축하여야 한다.
- 통합 테스트 결과서와 결함 관리 대장을 작성하여야 한다.
- 결함 조치가 완료되면 확인 테스트를 진행하여야 한다.

### 수행 순서

#### ① 작성된 통합 테스트 결과서를 검토한다.

통합 테스트 계획에 따라 수행된 테스트 결과서에 대해 테스트 케이스, 테스트 시나리오, 테스트 데이터, 예상 결과, 적합 여부, 심각도, 결함 내용 등을 검토한다.

#### ② 테스트 결과서의 결함 내용을 확인하고, 결함 관리 활동을 수행한다.

프로세스 단계별 결함에 대해 결함을 분석하고, 결함을 할당하고, 결함을 조치하는 등 결함을 관리한다.

##### 1. 결함이 발생한 테스트 산출물별 결함 ID를 부여한다.

결함이 발생한 통합 테스트에 대해 결함 조치 대상 및 결함 조치 현황을 추적할 수 있도록 결함 ID를 부여한다.

2. 부여된 결함 ID별 보고서를 기록한다.

발견된 결함이 실제 결함에 해당하는지 아니면 단순 에러인지 판단하고, 실제 결함일 경우 결함 관리 대장에 기록하고, 결함을 추적 관리한다.

- (1) 발견된 결함을 결함 관리 대장에 에러로 등록한다.
- (2) 등록된 결함을 분석하여 단순 결함인지, 실제 결함인지 확정한다.
- (3) 실제 결함으로 확정될 경우, 결함의 심각도와 우선순위를 설정한다.
- (4) 결함에 조치할 담당자를 배정하고, 조치 완료 예정일을 설정한다.

3. 결함 관리 대장을 작성하고 발생된 결함에 대해 추적 관리한다.

- (1) 프로세스 단계별 결함에 대해 결함 관리 대장을 작성한다.

분석 단계, 설계 단계, 테스트 단계별로 결함 관리 대장을 작성한다.

- (2) 각 단계별 결함 관리 대장에 기록할 항목들을 결정한 후 작성한다.

단계, 대상 산출물 명, 결함 ID, 결함 내용, 결함 유형, 심각도, 우선순위, 작성자, 결함 발생일, 결함 조치 대상, 조치 내용, 조치자, 확인자, 처리 완료일, 상태 등의 항목으로 결함 관리 대장을 완성한다.

<표 2-6> 결합 관리 대장 작성 예시

단계	대상 산출 물명	산출 물 식별 ID	결합 ID	결합 내용	결합 유형	심각 도	우선 순위	결합 발생 일	결합 조치 대상	조치 내용	조치 자	확인 인자	처리 완료 일	상태
테스 트공 정검 토	통합 테스트 결과 -xxx	DID_x xx_00 1	excel 과 불일치	DB	Normal	Low	2016. 01.05	프로 그램 수정	목록과 excel 처리 로직, SQL 확인	박 xx	전 xx	2016. 05.25	완 료	
테스 트공 정검 토	통합 테스트 결과 -xxx	DID_x xx_00 2	excel 과 불일치	DB	Normal	Low	2016. 01.05	프로 그램 수정	목록과 excel 처리 로직, SQL 확인	박 xx	전 xx	2016. 05.25	완 료	
테스 트공 정검 토	통합 테스트 결과 -xxx	DID_x xx_00 3	반복 처리 DB트랜잭션 요청	GUI	Normal	Medium	2016. 01.07	프로 그램 수정	Excute Batch로 변경	윤 xx	전 xx	2016. 02.24	완 료	
테스 트공 정검 토	통합 테스트 결과 -xxx	DID_x xx_00 4	삭제 요청 승인 시 SQL Exception 발생	기능	Major	Medium	2016. 01.08	프로 그램 수정	쿼리 검색조건 조건절 수정	강 xx	전 xx	2016. 02.24	완 료	
테스 트공 정검 토	통합 테스트 결과 -xxx	DID_x xx_00 5	엑셀다운로드 시관리 번호중복표시	기능	Minor	Medium	2016. 01.08	프로 그램 수정	엑셀 다운로드 시 관리번호 중복표시 수정	윤 xx	전 xx	2016. 02.24	완 료	
테스 트공 정검 토	통합 테스트 결과 -xxx	DID_x xx_00 6	일부자료에서 SQLException 발생	기능	Normal	Medium	2016. 01.08	프로 그램 수정	대상, 비대상에 대한 수량, 중량, SQL 수정 후 반영	김 xx	전 xx	2016. 02.24	완 료	

③ 작성된 결함 관리 대장의 내용을 확인하고, 결함을 분석 및 평가한다.

1. 발생한 결함을 유형별로 분류한다.

기능 오류, DB 오류, 인터페이스 오류, 콘텐츠 오류, 레이아웃 오류, 출력 내용 오류, 에러 처리 오류, 메시지 오류, GUI 오류, 기타 오류 등으로 분류한다.

2. 결함의 심각도와 발생한 빈도가 가장 높은 결함 유형을 파악한다.

결함 관리 대장의 결함 심각도 중 ‘단순 결함’인 경우와 ‘개선 권고’인 경우를 제외한 실제 결함에 대해 결함 유형을 파악한다.

3. 결함 분석을 위한 결함 관리 측정 지표를 설정한다.

지속적인 결함 관리를 위해 다음의 측정 지표를 추적 관리한다.

(1) 각 심각도별 미해결 결함의 개수

(2) 각 심각도별 일정 기간 내 수정된 결함의 개수

(3) 전체 발견된 결함의 개수

(4) 우선순위별 잔존 결함의 개수 및 심각도 수준

(5) 테스트 단계별 누적 결함의 개수

④ 발견된 결함에 대한 추이 분석을 하고, 잔존 결함에 대해 추정한다.

1. 결함 관리 측정 지표를 이용하여 결함의 추이를 분석한다.

(1) 특정 속성에 해당하는 결함 수를 측정하여 결함 분포를 분석한다.

(2) 시간 흐름에 따른 결함 수를 측정하여 결함 추세를 분석한다.

(3) 특정 결함 상태가 지속되는 시간을 측정하여 결함 에이징을 분석한다.

2. 추이 분석의 결과를 토대로 발생 가능한 잔존 결함에 대해 추정한다.

결함의 우선순위와 심각도를 고려하여 잔존 결함의 근원에 대해 추정한다.

### 수행 tip

- 통합 테스트를 수행하기 전에 테스트 계획서의 테스트 일정 및 테스트 조직에 대해 미리 확정한 후 테스트 환경을 구축한다.
- 테스트 수행 후 발견된 결함과 사용자가 발견한 결함을 더하여 총 결함 수로 설정하고 결함 분석을 수행하는 것이 좋다.

## 2-3. 애플리케이션 개선 조치사항 작성

### 학습 목표

- 개발자 통합 테스트 결과에 대한 분석을 통해 테스트의 충분성 여부를 검증하고, 발견된 결함에 대한 개선 조치사항을 작성할 수 있다.

### 필요 지식 /

#### ① 테스트 커버리지(Test Coverage)

##### 1. 개요

테스트 커버리지란 주어진 테스트 케이스에 의해 수행되는 소프트웨어의 테스트 범위를 측정하는 테스트 품질 측정 기준이며, 테스트의 정확성과 신뢰성을 향상시키는 역할을 한다.

##### 2. 기능 기반 커버리지

테스트 대상 애플리케이션의 전체 기능을 모수로 설정하고, 실제 테스트가 수행된 기능의 수를 측정하는 방법이다. 기능 기반 테스트 커버리지는 100% 달성을 목표로 하며, 일반적으로 UI가 많은 시스템의 경우 화면 수를 모수로 사용할 수도 있다.

##### 3. 라인 커버리지(Line Coverage)

애플리케이션 전체 소스 코드의 Line 수를 모수로 테스트 시나리오가 수행한 소스 코드의 Line 수를 측정하는 방법이다. 단위 테스트에서는 이 라인 커버리지를 척도로 삼기도 한다.

##### 4. 코드 커버리지(Code Coverage)

소프트웨어 테스트 충분성 지표 중 하나로 소스 코드의 구문, 조건, 결정 등의 구조 코드 자체가 얼마나 테스트되었는지를 측정하는 방법이다.

###### (1) 구문(Statement) 커버리지

코드 구조 내의 모든 구문에 대해 한 번 이상 수행하는 테스트 커버리지를 말한다.

###### (2) 조건(Condition) 커버리지

결정 포인트 내의 모든 개별 조건식에 대해 수행하는 테스트 커버리지를 말한다.

###### (3) 결정(Decision) 커버리지

결정 포인트 내의 모든 분기문에 대해 수행하는 테스트 커버리지를 말한다.

###### (4) 변형 조건/결정(Modified Condition/Decision) 커버리지

조건과 결정을 복합적으로 고려한 측정 방법이며, 결정 포인트 내의 다른 개별적인 조건식 결과에 상관없이 독립적으로 전체 조건식의 결과에 영향을 주는 테스트 커버리지를 말한다.

## ② 테스트 결함 식별 및 관리

### 1. 결함의 식별

#### (1) 단계별 결함 유입 분류

##### (가) 기획 시 유입되는 결함

사용자 요구사항의 표준 미준수로 인해 테스트 불가능, 요구사항 불명확/불완전/불일치, 기타 결함이 발생할 수 있다.

##### (나) 설계 시 유입되는 결함

기획 단계에 유입된 결함 또는 설계 표준 미준수로 인해 테스트 불가능, 기능 설계 불명확/불완전/불일치, 기타 결함이 발생할 수 있다.

##### (다) 코딩 시 유입되는 결함

설계 단계에 유입된 결함 또는 코딩 표준 미준수로 인해 기능의 불일치/불완전, 데이터 결함, 인터페이스 결함, 기타 결함이 발생할 수 있다.

##### (라) 테스트 부족으로 유입되는 결함

테스트 수행 시 테스트 완료 기준의 미준수, 테스트 팀과 개발 팀의 의사소통 부족, 개발자의 코딩 실수로 인한 결함이 발생할 수 있다.

#### (2) 결함 심각도별 분류

애플리케이션에 발생한 결함이 어떤 영향을 끼치며, 그 결함이 얼마나 치명적인지를 나타내는 척도이다.

##### (가) 결함 심각도 분류 사례

치명적(Critical) 결함, 주요(Major) 결함, 보통(Normal) 결함, 경미한(Minor) 결함, 단순(Simple) 결함 등으로 분류할 수 있다.

##### (나) 결함 심각도 관리

결함 관리의 정확성과 신뢰성 향상을 위해 결함 심각도의 각 단계별로 표준화된 용어를 사용하여 정의하여야 하며, 프로젝트 및 조직 차원에서 결함 관리 활동을 수행해야 한다.

#### (3) 결함 우선순위

##### (가) 결함 심각도와 결함 우선순위

결함 우선순위는 발생한 결함이 얼마나 빠르게 처리되어야 하는지를 결정하는 척도로, 결함 심각도가 높아도 우선순위가 반드시 높은 것은 아니며, 애플리케이션의 특성에 따라 우선순위가 결정될 수 있다.

#### (나) 결함 우선순위 표현 사례

결정적(Critical), 높음(High), 보통(Medium), 낮음(Low) 또는 즉시 해결, 주의 요망, 대기, 개선 권고 등으로 분류할 수 있다.

### 2. 결함 관리 항목

테스트 수행 후 발견된 결함은 결함 관리 시스템에 등록하여 관리해야 하며, 등록 시 다음 항목들은 필수로 등록한다.

- (1) 결함 내용
- (2) 결함 ID
- (3) 결함 유형
- (4) 발견일
- (5) 심각도
- (6) 우선순위(결함 수정의 우선순위)
- (7) 시정 조치 예정일
- (8) 수정 담당자
- (9) 재테스트 결과
- (10) 종료일

## 수행 내용 / 애플리케이션 개선 조치사항 작성하기

---

### 재료 · 자료

- SW 개발 국제 표준(ISO/IEC 9126, ISO/IEC 14598, ISO15504, ISO/IEC 12207, CMMI 등)
- 소프트웨어 테스트 관련 국제 표준(ISO/IEC 29119), TMMI(Test Maturity Model integration) Model
- 통합 테스트 계획서, 테스트 결과서, 결함 관리 대장, 테스트 분석서

### 기기(장비 · 공구)

- 전산 장비(컴퓨터, 프린터, 빔 프로젝터, 네트워크 장비 등)
- OA 도구(워드프로세서, 스프레드시트, 프레젠테이션 도구 등)
- CASE 도구(SW설계 지원 도구, SW 개발 지원 도구, SW 테스트 지원 도구 등)

## 안전 · 유의 사항

- 테스트 커버리지의 종류에 대해 학습하고, 테스트 커버리지를 평가하여야 한다.
- 사용자 요구사항을 추적할 수 있는 요구사항 추적 매트릭스를 작성한다.
- 테스트 케이스 성공률을 미리 계획하고, 테스트 수행 통계를 작성하여야 한다.
- 테스트 수행 후 발견한 결함에 대해 결함을 수정하고 조치 이력을 관리하여야 한다.

## 수행 순서

### ① 통합 테스트 결과에 대해 상세 분석한다.

테스트 수행 결과로 작성된 모든 테스트 결과 보고서와 결함 관리 대장을 검토한다.  
(‘<표 2-4> 통합 테스트 결과서 작성 예시’ 및 ‘<표 2-6> 결함 관리 대장 작성 예시’  
참조)

### ② 테스트 커버리지를 평가한다.

사용자 요구사항에 대해 모든 기능에 대하여 테스트를 100% 수행할 수 없으므로, 테스트 수행 정도를 파악하기 위해 테스트 커버리지를 측정한다.

#### 1. 테스트 커버리지 요건 달성을 계획을 수립한다.

##### (1) 코드 커버리지의 종류에 대해 파악하고, 정리한다.

(가) 제어 흐름 기반 테스트 커버리지의 종류를 파악한다.

구문, 결정, 조건, 조건/결정, 변형 조건/결정, 다중 조건 등의 커버리지에 대해 파악한다.

(나) 자료 흐름 기반 테스트 커버리지의 종류를 파악한다.

All Node, All Edge, All Defs, All Uses, All du paths 등 커버리지를 파악한다.

##### (2) 요구사항 커버리지에 대해 파악한다.

사용자 요구사항 정의서의 기능을 대상으로 얼마나 테스트되었는지를 파악한다.

##### (3) 소프트웨어 중요도에 따른 테스트 커버리지 요건 및 검증 계획을 수립한다.

문장 테스트 커버리지를 95% 이상 수행해야 한다는 요건을 수립한다.

#### 2. 테스트 커버리지를 측정하고, 분석을 통해 요건 만족을 검증한다.

##### (1) 코드 커버리지를 측정한다.

##### (2) 요구사항 커버리지를 측정하고 요구사항 추적 매트릭스(RTM : Requirement Traceability Matrix)를 작성한다.

(3) 커버리지 요건 만족 여부를 검증한다.

테스트 케이스를 나열하고, 테스트 수행 결과서를 검토하여 커버리지 요건 만족 여부를 분석 및 검증한다.

<표 2-7> 기능 요구사항 추적 매트릭스 작성 예시

③ 테스트 케이스 통계 및 분석을 통해 테스트 충분성 여부를 파악한다.

테스트 케이스 검증률과 성공률을 파악하여 테스트 충분성 여부를 검증한다.

1. 테스트 수행 통계를 바탕으로 테스트 케이스 검증률을 파악한다.

시간의 흐름에 따른 테스트 검증 요청 건수에 대해 통계를 작성하고, 검증률 추이를 분석한다.

2. 테스트 수행 통계를 바탕으로 테스트 케이스 성공률을 파악한다.

일정 기간 동안의 테스트 성공률에 대한 통계를 작성하고, 검증률 추이를 분석한다.

3. 테스트 케이스 검증률과 테스트 케이스 성공률을 통해 테스트 충분성을 검증한다.

④ 발견된 결함에 대한 개선 조치사항을 작성한다.

1. 결함 개선 조치 계획을 수립한다.

발견된 결함의 심각도와 애플리케이션의 특성에 따라 개선할 우선순위를 결정한다.

2. 우선순위에 따라 결함을 조치할 담당자를 할당한다.

3. 발견된 결함에 대해 프로그램을 개선하고 조치한다.

⑤ 결합 조치 이력을 관리한다.

1. 조치 완료된 결함에 대해 조치 사항을 검토한다.

수정된 결함에 대해 회기 테스트 또는 확인 테스트를 수행하여 결함이 정상적으로 조치되었는지를 확인한다.

2. 정상적으로 조치 완료된 결합 ID에 대한 상태를 결합 관리 대장에 기록한다.

정상적으로 조치가 완료된 테스트 항목에 대해 결합 관리 대장에 상태를 ‘완료’로 기록하고 지속적으로 관리한다.

3. 결합 조치 완료에 대해 승인한다.

품질 관리자가 수시 모니터링을 통하여 결합 조치가 완료된 항목에 대해 확인한 후 ‘확인 완료’로 상태 항목을 설정한다.

### 수행 tip

- 사용자 요구사항 중 기능 요구사항의 구현 여부를 검증하기 위한 요구사항 추적 매트릭스를 작성하고 검증 방안으로 테스트 수행 결과를 기록한다.
- 발견된 결함에 대해 누가, 언제, 어떤 부분을 수정할 것인지 결합 수정 계획을 구체적으로 수립하여야 한다.

## 학습2 교수 · 학습 방법

### 교수 방법

- 학습자의 통합 테스트 방법과 테스트 절차에 대한 지식 습득 여부를 확인하고, 수업을 진행한다.
- 통합 모듈 및 인터페이스가 사용자 요구사항을 어떻게 검증하는지를 설명하고, 학습자의 이해를 바탕으로 실습을 진행한다.
- 결함 관리 대장 작성 시 조치 단계를 세분화하여 작성하고, 결함의 심각도와 우선순위를 선정하고, 조치 담당자를 선정하여 완료 예정일을 확정할 수 있도록 지도한다.
- 통합 테스트 결과 분석을 통해 결함 추이 분석을 하고, 이를 통해 잔존 결함을 추정할 수 있도록 설명한 후 수업을 진행한다.
- 통합 테스트 결과 분석을 통해 테스트의 충분성 여부를 검증하는 방법과 결함 개선 조치 계획을 작성하는 방법에 대해 설명한 후 수업을 진행한다.

### 학습 방법

- 통합 테스트 방법과 테스트 절차에 대한 지식을 습득하고, 실습한다.
- 통합 모듈 및 인터페이스가 사용자 요구사항을 어떻게 검증하는지를 이해하고, 실습한다.
- 결함 조치 단계를 세분화하여 작성하고, 결함의 심각도와 우선순위를 선정하고, 조치 담당자를 선정하고, 완료 예정일을 확정하여 결함 관리 대장을 작성한다.
- 통합 테스트 결과 분석을 통해 결함 추이 분석을 하고, 이를 통해 잔존 결함을 추정하는 방안에 대해 이해한 후 실습한다.
- 통합 테스트 결과 분석을 통해 테스트의 충분성 여부에 대한 검증 방안을 파악하고, 결함 개선 조치 계획을 작성할 수 있도록 실습한다.

## 학습2 평 가

### 평가 준거

- 평가자는 학습자가 학습 목표를 성공적으로 달성하였는지를 평가해야 한다.
- 평가자는 다음 사항을 평가해야 한다.

학습 내용	학습 목표	성취수준		
		상	중	하
애플리케이션 통합 테스트 수행	- 개발자 통합 테스트 계획에 따라 통합 모듈 및 인터페이스가 요구사항을 충족하는지에 대한 테스트를 수행 할 수 있다.			
애플리케이션 테스트 결과 분석	- 개발자 통합 테스트 수행 결과 발견된 결함에 대한 추이 분석을 통하여 잔존 결함을 추정할 수 있다.			
애플리케이션 개선 조치사항 작성	- 개발자 통합 테스트 결과에 대한 분석을 통해 테스트의 충분성 여부를 검증하고, 발견된 결함에 대한 개선 조치사항을 작성할 수 있다.			

### 평가 방법

- 문제해결 시나리오

학습 내용	평가 항목	성취수준		
		상	중	하
애플리케이션 통합 테스트 수행	- 통합 테스트 절차에 대한 이해 능력과 통합 테스트 계획서 작성 능력 - 다양한 방식의 통합 테스트 수행 능력			
애플리케이션 테스트 결과 분석	- 통합 테스트 결과 작성 능력 및 결함 식별 능력 - 결함 추이 분석 능력			
애플리케이션 개선 조치사항 작성	- 결함 식별 및 조치 우선순위 결정 능력 - 결함 조치 이력 관리 능력			

• 평가자 체크리스트

학습 내용	평가 항목	성취수준		
		상	중	하
애플리케이션 통합 테스트 수행	<ul style="list-style-type: none"> <li>- 작성된 통합 테스트 계획서의 완성도</li> <li>- 통합 테스트 수행 방안의 이해 능력</li> </ul>			
애플리케이션 테스트 결과 분석	<ul style="list-style-type: none"> <li>- 작성된 통합 테스트 결과서의 완성도</li> <li>- 결함 추이 분석을 위한 결함 측정 항목의 이해 능력</li> </ul>			
애플리케이션 개선 조치사항 작성	<ul style="list-style-type: none"> <li>- 결함의 심각도 및 우선순위 이해 능력</li> <li>- 결함 조치 이력 관리 항목의 이해 정도</li> </ul>			

## 피드백

### 1. 문제해결 시나리오

- 통합 테스트 계획서를 검토하고, 테스트 케이스 및 시나리오에 따라 통합 테스트의 수행 능력을 평가하고, 통합 테스트 결과를 작성하고 식별된 결함에 대해 추이 분석 및 결함 관리 능력을 평가하여 일정 수준 미달인 학습자에게 통합 테스트 수행 및 결과 관리와 관련한 추가 학습을 통하여 다시 테스트할 수 있도록 한다.

### 2. 평가자 체크리스트

- 통합 테스트 계획, 수행 방안, 통합 테스트 결과, 결함 측정 항목의 이해, 결함의 심각도 및 우선순위 이해, 결함 조치 이력 관리 항목에 대한 이해 정도를 평가한 후 일정 수준 이하인 학습자에게 추가 학습을 진행하고 각 항목에 대한 이해 능력을 재평가한다.

학습 1	애플리케이션 테스트 케이스 설계하기
학습 2	애플리케이션 통합 테스트하기

## 학습 3 애플리케이션 성능 개선하기

### 3-1. 애플리케이션 성능 분석

#### 학습 목표

- 애플리케이션 테스트를 통하여 애플리케이션의 성능을 분석하고, 성능 저하 요인을 발견할 수 있다.

#### 필요 지식 /

##### ① 애플리케이션 성능 점검의 개요

애플리케이션 성능이란 사용자의 요구 기능을 해당 애플리케이션이 최소의 자원을 사용하면서 얼마나 빨리, 많은 기능을 수행하는가를 육안 또는 도구를 통하여 점검하는 것을 말한다.

##### 1. 애플리케이션의 성능을 측정하기 위한 지표

###### (1) 처리량(Throughput)

애플리케이션이 주어진 시간에 처리할 수 있는 트랜잭션의 수로, 웹 애플리케이션의 경우 시간당 페이지 수로 표현하기도 한다.

###### (2) 응답 시간(Response Time)

사용자 입력이 끝난 후, 애플리케이션의 응답 출력이 개시될 때까지의 시간으로, 웹 애플리케이션의 경우 메뉴 클릭 시 해당 메뉴가 나타나기까지 걸리는 시간을 말한다.

###### (3) 경과 시간(Turnaround Time)

애플리케이션에 사용자가 요구를 입력한 시점부터 트랜잭션 처리 후 그 결과의 출력이 완료할 때까지 걸리는 시간을 말한다.

###### (4) 자원 사용률(Resource Usage)

애플리케이션이 트랜잭션 처리하는 동안 사용하는 Cpu 사용량, 메모리 사용량, 네트워크 사용량을 말한다.

##### 2. 유형별 성능 분석 도구

성능 분석 도구는 애플리케이션의 성능을 점검하는 도구와 시스템 자원 사용량을 모니터링하는 도구로 분류할 수 있다.

### (1) 성능/부하/스트레스(Performance/Load/Stress) 점검 도구

애플리케이션의 성능 점검을 위해 가상의 사용자를 점검 도구 상에서 인위적으로 생성한 뒤, 시스템의 부하나 스트레스를 통해 성능 측정 지표인 처리량, 응답 시간, 경과 시간 등을 점검하기 위한 도구이다.

### (2) 모니터링(Monitoring) 도구

모니터링 도구는 애플리케이션 실행 시 자원 사용량을 확인하고 분석 가능한 도구로, 성능 모니터링, 성능 저하 원인 분석, 시스템 부하량 분석, 장애 진단, 사용자 분석, 용량 산정 등의 기능을 제공하여, 시스템의 안정적 운영을 지원하는 도구이다.

## ② 애플리케이션 성능 저하 원인 분석

애플리케이션의 성능이 저하되는 원인은 크게 DB 연결 및 쿼리 실행, 내부적인 요인과 외부적인 요인, 그리고 기타 환경 설정이나, 네트워크 등의 문제로 구분할 수 있다.

### 1. 데이터베이스 연결 및 쿼리 실행 시 발생되는 성능 저하 원인

일반적으로 DB에 연결하기 위해 Connection 객체를 생성하거나 쿼리를 실행하는 애플리케이션 로직에서 성능 저하 또는 장애가 많이 발견된다.

#### (1) 데이터베이스 락(DB Lock)

대량의 데이터 조회, 과도한 업데이트, 인덱스 생성 시 발생하는 현상이며, 요청한 작업은 Lock의 해제 시까지 대기하거나 타임아웃된다.

#### (2) 불필요한 데이터베이스 페치(DB Fetch)

실제 필요한 데이터보다 많은 대량의 데이터 요청이 들어올 경우, 또는 결과 세트에서 마지막 위치로 커서를 옮기는 작업이 빈번한 경우 응답 시간 저하 현상이 발생한다.

#### (3) 연결 누수(Connection Leak) / 부적절한 커넥션 풀 크기(Connection Pool Size)

##### (가) 연결 누수(Connection Leak)

DB 연결과 관련한 JDBC 객체를 사용 후 종료하지 않을 경우 발생한다.

##### (나) 부적절한 커넥션 풀 크기(Connection Pool Size)

너무 작거나 크게 설정한 경우 성능 저하 현상이 발생할 가능성이 있다.

##### (4) 기타

트랜잭션이 확정(Commit)되지 않고 커넥션 풀에 반환되거나, 잘못 작성된 코드로 인해 불필요한 Commit가 자주 발생하는 경우 성능이 저하될 가능성이 존재한다.

### 2. 내부 로직으로 인한 성능 저하 원인

#### (1) 웹 애플리케이션의 인터넷 접속 불량

웹 애플리케이션의 실행 시 인터넷 접속 불량으로 서버 소켓(Server Socket) 쓰기는 지속되나, 클라이언트에서 정상적 읽기가 수행되지 않아 성능이 저하될 가능성이 있다.

### (2) 특정 파일의 업로드, 다운로드로 인한 성능 저하

대량의 파일을 업로드하거나 다운로드할 경우 처리시간이 길어져 애플리케이션의 성능이 저하될 가능성이 있다.

### (3) 정상적으로 처리되지 않은 오류 처리로 인한 성능 저하

오류 처리 로직과 실제 처리 로직 부분을 분리하지 않고 코딩하거나 예외가 발생할 경우에 제대로 처리되지 않아 행이 걸리는 상황이 발생하여 성능이 저하될 가능성이 있다.

## 3. 외부 호출(HTTP, 소켓 통신)로 인한 성능 저하 원인

임의의 트랜잭션이 수행되는 동안 외부 트랜잭션(외부 호출)이 장시간 수행되거나, 타임아웃이 일어나는 경우 성능 저하 현상이 발생할 수 있다.

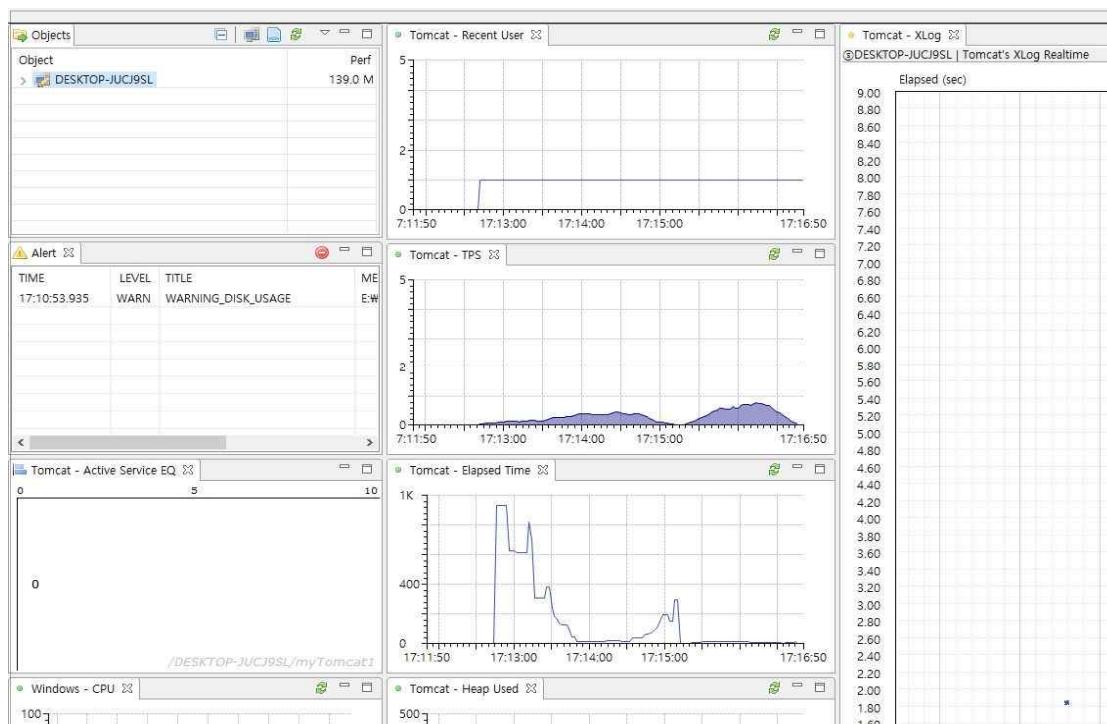
## 4. 잘못된 환경 설정이나 네트워크 문제로 인한 성능 저하 원인

### (1) 환경 설정으로 인한 성능 저하

스레드 풀(Thread Pool), 힙 메모리(Heap Memory)의 크기를 너무 작게 설정하면 Heap Memory Full 현상 발생으로 성능이 저하될 가능성이 있다.

### (2) 네트워크 장비로 인한 성능 저하

라우터, L4 스위치 등 네트워크 관련 장비 간 데이터 전송 실패 또는 전송 지연에 따른 데이터 손실 발생 시 애플리케이션의 성능 저하 또는 장애가 발생할 수 있다.



[그림 3-1] 시스템 모니터링 화면 예시(Scouter 활용)

## 수행 내용 / 애플리케이션 성능 분석하기

---

### 재료 · 자료

- SW 개발 국제 표준(ISO/IEC 9126, ISO/IEC 14598, ISO15504, ISO/IEC 12207, CMMi 등)
- 소프트웨어 테스트 관련 국제 표준(ISO/IEC 29119)
- 요구사항 정의서, 시스템 테스트 계획서, 성능 점검 도구, 성능 모니터링 도구

### 기기(장비 · 공구)

- 전산 장비(컴퓨터, 프린터, 빔 프로젝터, 네트워크 장비 등)
- OA 도구(워드프로세서, 스프레드시트, 프레젠테이션 도구 등)
- 성능 분석 도구(SW 설계 지원 도구, 성능 측정 지원 도구 등)

### 안전 · 유의 사항

- 애플리케이션 성능 점검을 위한 방법과 절차에 따라 학습하여야 한다.
- 성능 테스트 도구 및 모니터링 도구의 사용 방법과 유의 사항을 습득하여야 한다.
- 성능 점검을 위한 성능 점검 계획서와 성능 테스트 시나리오를 작성하여야 한다.
- 애플리케이션의 성능을 테스트하기 위한 서버는 실제 사용 중인 장비를 활용할 수 있도록 준비하여야 한다.

### 수행 순서

#### ① 애플리케이션 성능 점검을 위한 도구의 유형을 정리한다.

애플리케이션 성능 점검을 위한 성능 테스트 도구 및 시스템 모니터링 도구의 유형을 파악하고 그 특징을 간략하게 정리한다.

##### 1. 성능 테스트 도구의 유형에 대해 파악한다.

성능 테스트 도구의 도구 명, 도구 설명, 지원 환경, 지원 홈페이지 등의 정보를 파악하고 정리한다.

<표 3-1> 오픈소스 성능 테스트 도구

도구 명	도구 설명	지원환경	홈페이지
JMeter	HTTP, FTP, LDAP 등 다양한 프로토콜 지원하는 안전성, 확장성, 부하, 기능 테스트 도구	Cross-Platform	<a href="http://jmeter.apache.org/">http://jmeter.apache.org/</a>
LoadUI	HTTP, JDBC 등 주로 웹서비스를 대상으로 서버모니터링을 지원하는 UI를 강화한 부하 테스트 도구	Cross-Platform	<a href="https://www.loadui.org/">https://www.loadui.org/</a>
OpenSTA	HTTP, HTTPS 지원하는 부하 테스트 및 생산품 모니터링 도구	Windows	<a href="http://opensta.org/">http://opensta.org/</a>

## 2. 시스템 모니터링 도구의 유형에 대해 파악한다.

시스템 모니터링 도구의 도구 명, 도구 설명, 지원 환경, 개발 도구 지원 여부, 지원 홈페이지 등의 정보를 파악하고 확인한다.

<표 3-2> 오픈소스 시스템 모니터링 도구

도구 명	도구 설명	지원환경	홈페이지
Scouter	단일 뷰 통합/실시간 모니터링, 튜닝에 최적화된 인프라 통합 모니터링 도구	Cross-Platform	<a href="https://github.com/scouter-project/scouter">https://github.com/scouter-project/scouter</a>
Zabbix	웹기반 서버, 서비스, 애플리케이션 모니터링 도구	Cross-Platform	<a href="https://www.zabbix.com/">https://www.zabbix.com/</a>

## ② 애플리케이션 성능 점검 계획서를 작성한다.

### 1. 성능 점검 계획서에 포함될 항목에 대해 검토한다.

애플리케이션 성능을 측정하기 위한 성능 점검 계획서의 각 항목에 대해 정의한다.

#### (1) 성능 점검 목적 및 용어를 정의한다.

해당 시스템의 성능 점검을 수행하는 목적 및 성능 점검 시 활용할 용어에 대해 정의 한다.

#### (2) 성능 점검 수행 전략을 수립한다.

##### (가) 대상 시스템의 구성도를 파악한다.

전체 시스템 중 성능을 점검해야 할 시스템을 파악하여 대상 구성도를 작성한다.

(나) 대상 서버의 정보를 파악한다.

대상 서버의 서버 명, IP주소, 설치된 운영 체제, CPU, 메모리, 설치된 솔루션 정보를 파악하여 작성한다.

(다) 성능 점검 수행 도구를 결정한다.

성능 점검을 위한 오픈소스 또는 상업용 수행 도구를 결정하여 작성한다.

(라) 성능 점검 환경을 구성한다.

성능 점검을 위한 점검 팀을 확정하고, 부하 발생 장비, 부하 발생 환경에 대하여 확인한다.

(3) 성능 점검 수행 일정 및 절차에 대해 확정한다.

(가) 성능 점검 수행 일정에 대해 확정한다.

계획 수립, 설계 및 개발, 테스트 환경 구축, 테스트 수행, 결과 분석, 보고서 작성 등에 대한 성능 점검 수행 일정에 대해 확정한다.

(나) 성능 점검 수행 절차에 대해 확정한다.

테스트 계획 수립, 테스트 시나리오 작성, 테스트 케이스 작성, 워크로드(Workload) 설계, 레코딩, 테스트 수행, 결과 분석 등 성능 점검 수행 절차를 확정한다.

(다) 성능 개선 절차에 대해 확정한다.

기본 성능 자료를 도출하고, 성능 테스트 결과를 분석하고, 개선안을 적용하고, 겸 중하여 성능 개선을 검증하는 절차에 대해 확정한다.

(라) 성능 점검 역할별 수행 인력을 확정한다.

성능 팀, 공통 테스트 팀, 업무 개발 팀의 팀별 수행 인력을 확정한다.

(4) 성능 점검 수행 방안에 대해 작성한다.

(가) 각 시스템의 단위 성능 테스트 방법에 대해 결정한다.

성능 테스트의 경우 가상 인원이 몇 명인지, Think Time(서버로부터 응답 후, 다음 동작 때까지 대기하는 시간)의 경우 몇 초로 테스트할 것인지를 결정한다.

(나) 시스템 성능 목표를 설정한다.

각 시스템별 테스트 유형과 대상 업무, 목표 부하, 테스트 시간 등을 확정하고 시스템의 성능 목표를 설정한다.

(다) 모니터링 및 성능 지표 수집 방안에 대해 결정한다.

측정 대상을 구분하고 측정 목적, 시스템 명, 부하량, 배치 작업 소요 시간, SQL 등 의 모니터링 방안, 모니터링 측정 항목에 대해 설정한다.

(라) 성능 테스트 시나리오를 작성한다.

Ramp-Up Model, Ramp-Down Model, Think Time Model 등 다양한 성능 테스트 모델별 성능 테스트 시나리오를 작성한다.

## 2. 성능 테스트 계획서를 작성한다.

성능 점검 개요, 성능 점검 수행 전략, 성능 점검 수행 일정 및 절차, 성능 점검 수행 방안 등 결정된 항목을 포함하여 성능 테스트 계획서를 작성한다.

xx 시스템 제 n 차 확대구축사업 (xx 부문)		성능점검계획서
<u>목차</u>		
1. 성능점검 개요 .....	.....	3
1.1 목적 .....	.....	3
2. 성능점검 수행 전략 .....	.....	4
2.1 대상 구성도 .....	.....	4
2.2 대상 서버 .....	.....	4
2.3 성능점검 솔루션 .....	.....	5
2.4 성능점검 환경 .....	.....	5
3. 성능점검 수행 일정 및 절차 .....	.....	5
3.1 수행 일정 .....	.....	5
3.2 수행 절차 .....	.....	6
3.3 개선 절차 .....	.....	6
4. 성능점검 수행 방안 .....	.....	7
4.1 단위 성능 테스트 방법 .....	.....	7
4.2 시각화 성능 점검 .....	.....	7
4.3 배치작업 성능 점검 .....	.....	7
4.4 시각화 및 배치작업 성능 목표 .....	.....	8
4.5 시스템 성능 목표 .....	.....	8
4.6 모니터링 및 성능지표 수집 방안 .....	.....	8
4.7 테스트 시나리오 .....	.....	9

[그림 3-2] 성능 테스트 계획서 목차 예시

### ③ 애플리케이션 성능 테스트를 수행한다.

애플리케이션 성능 점검 목표에 따라 성능 테스트를 수행한다.

#### 1. 애플리케이션 성능 테스트 케이스를 작성한다.

애플리케이션 성능 측정을 위한 테스트 케이스를 아래 항목을 포함하여 작성한다.

##### (1) 테스트 목표 및 목표 값을 설정하여 작성한다.

테스트 상황 및 사용자 수, 호출 간격, TPS(Transaction Per Second), 응답 시간 등 목표 값을 설정한다.

##### (2) 측정 항목을 기술하여 작성한다.

TPS, 응답 시간, 시스템 사용률, 거래 성공 비율 등 측정 항목에 대해 기술한다.

##### (3) 테스트 시나리오를 작성한다.

성능 테스트에 대한 구체적인 방법 및 절차에 대해 작성한다.

##### (4) 사전 확인 사항에 대해 작성한다.

테스트 시작 시간, 종료 시간, 스크립트 수행 횟수, 부하 발생기 상태 확인, 데이터베이스 상태 확인, 투입인력 확인, 테스트 환경 설정, 테스트 데이터 등에 대해 작성한다.

<표 3-3> 애플리케이션 성능 테스트 케이스 작성 예시

<b>테스트 목표</b>	성능 부하 시험 테스트로써 처리량 16.7 TPS 이상		
<b>목표치</b>	vUser	20	호출간격 10초
	TPS	> 16.7	응답시간 3초 이내
<b>성능 지표</b>	처리량(TPS:Pages per Second) 응답시간(Response Time, 평균, 90th) 자원 사용률(System Resource, Cpu Usage, Memory Usage)		
<b>시나리오</b>	<ul style="list-style-type: none"> <li>- Fixed Mode</li> <li>Run time : 5 분, Number of vUsers : 1 user</li> <li>- Ramp-up Mode</li> <li>Run time : 10 분 : 1 ~ 5 분(Ramp-up 구간), 5 ~ 10 분(Fix 구간)</li> <li>Test page_01 : <a href="http://www.xxx.kr/001">http://www.xxx.kr/001</a> ~ page_15 : <a href="http://www.xxx.kr/015">http://www.xxx.kr/015</a></li> </ul>		
<b>사전확인사항</b>	테스트 시작 시간	테스트 종료 시간	
	스크립트 수행(N번)	부하 발생기 상태	
	데이터베이스 상태	투입인력 확인	
	테스트 환경 설정	테스트 데이터	

## 2. 애플리케이션 성능 테스트를 수행한다.

작성된 테스트 케이스 및 테스트 시나리오에 따라 애플리케이션 성능 테스트를 수행한다.

### (1) 선정된 성능 테스트 도구를 설치한다.

대상 시스템에 선정된 테스트 도구를 설치한다.

### (2) 테스트 도구의 테스트 환경을 설정한다.

해당 시스템의 운영 체제, DBMS 버전, 네트워크 상태 등에 대해 설정한다.

### (3) 성능 테스트를 위한 시나리오를 생성한다.

테스트 목적에 맞는 Load Type, 파라미터, 사용자 수, Ramp-up load, Periodic load, 수행 시간, 모니터링 결과 저장 파일 등의 정보를 설정한다.

### (4) 시나리오를 실행하면서 테스트 상황을 모니터링한다.

성능 테스트를 수행하면서 테스트 상황을 도구를 통해 모니터링한다.

④ 성능 테스트 결과 분석을 통해 성능 저하 요인을 발견한다.

애플리케이션의 성능 테스트 결과를 분석하여 성능 저하 요인을 발견한다.

1. 애플리케이션의 성능 테스트 결과를 분석한다.

애플리케이션 성능 목표 대비 결과를 분석한다.

(1) 사용자 수 증가에 따른 트랜잭션 성공 비율에 대해 추이를 분석한다.

(2) TPS, 응답 시간에 대해 분석한다.

(3) 시스템 자원 사용률에 대해 분석한다.

CPU 사용률, 메모리 사용률, DB 사용률에 대해 분석한다.

2. 애플리케이션의 성능 저하 요인을 발견하고, 분석한다.

응답 시간이 목표 이하로 발생되는 애플리케이션의 성능 저하 요인을 분석한다.

(1) 애플리케이션 성능 부하 테스트 결과에서 성능 저하 요인을 발견한다.

각 애플리케이션별 가상 사용자, TPS, 목표 TPS, 응답 시간 평균, 응답 시간 90% 등의 데이터에서 목표를 달성하지 못한 애플리케이션을 발견하고, 저하 요인을 분석한다.

(2) 응답 시간 측정 및 분석 결과에서 성능 저하 요인을 발견한다.

대상 업무별 응답 시간 평균, 응답 시간 90% 측정 결과 성능 목표 미달성 애플리케이션의 성능 저하 요인을 분석한다.

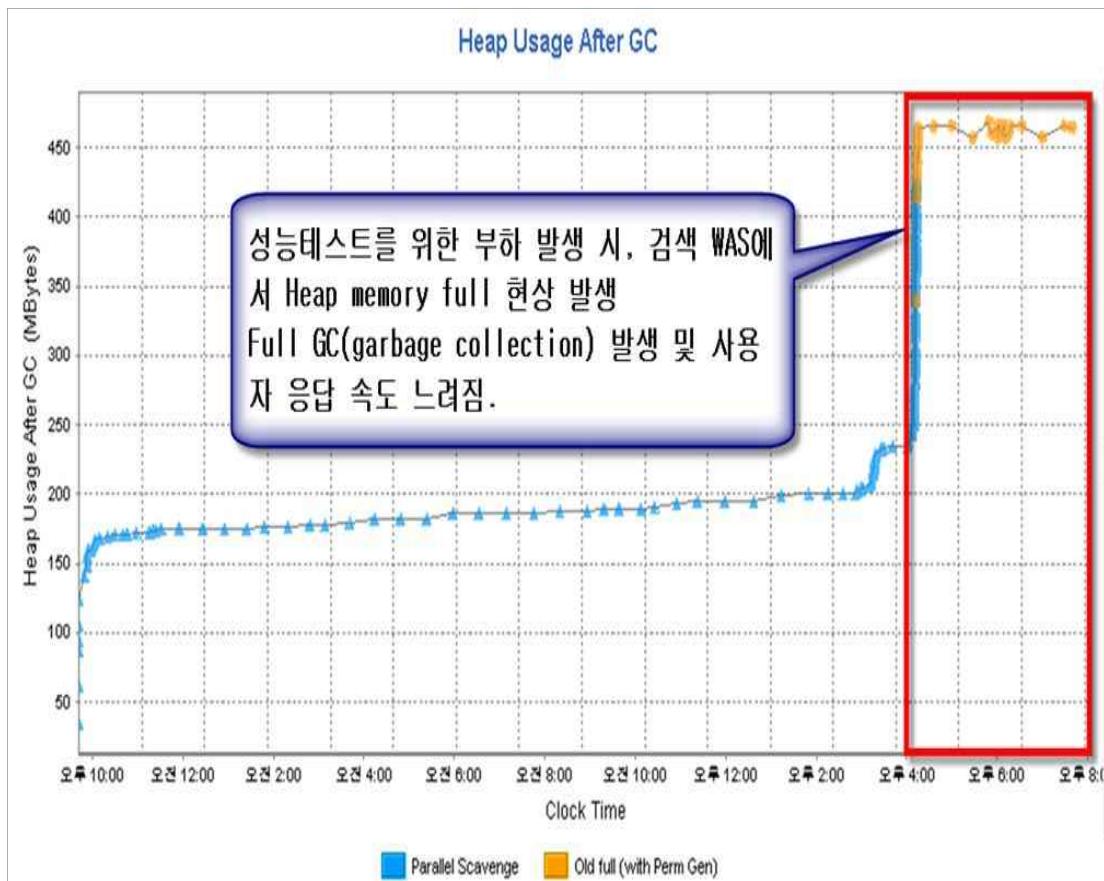
[표 8] 성능부하테스트 상세 내역

시스템명	테스트케이스 ID	업무명	응답시간 평균	응답시간 90%	vUser	TPS	목표 TPS
xx분석 포털	A-XX01_TC01	로그인페이지	0.07	0.25	18	9.55	3.33
	A-XX01_TC02	로그인	1.80	5.00			
	A-XX01_TC03	게시판	1.15	3.04			
	A-XX01_TC04	게시판 조회	0.41	0.76			
	A-XX01_TC05	로그아웃	0.10	0.37			
EDMS	A-XX02_TC01	xx 메인 화면	1.09	2.41	16	9.92	3.33
	A-XX02_TC02	xx보고서	0.84	1.36			
시맨틱 통합검색	A-XX03_TC01	검색	0.69	1.21	16	6.19	3.33
	A-XX03_TC02	검색 - 인물선택	2.24	3.96			
소셜 네트워크	A-XX04_TC01	xx검색	0.20	0.38	16	17.20	3.33
	A-XX04_TC02	소셜네트워크 분석	0.30	1.47			
xx 분석	A-XX05_TC01	자동완성	0.00	0.01	18	163.64	3.33
	A-XX05_TC02	검색결과	0.15	0.17			
	A-XX05_TC03	xx분석결과	0.11	0.14			
xx 인물관리	A-XX06_TC01	xx인물 메인 화면	0.84	1.62	18	4.59	3.33
	A-XX06_TC02	조회	0.60	1.25			
	A-XX06_TC03	인물선택	5.91	8.89			

[그림 3-3] 성능 테스트 결과 예시

(3) 장애 또는 성능 저하 요인에 대해 원인을 분석한다.

시스템 장애, 응답 시간 저하 등의 원인에 대해 분석한다.



[그림 3-4] 성능 저하 요인 분석 예시

### 수행 tip

- 성능 테스트 도구와 성능 모니터링 도구 선정 시 오픈소스 라이선스를 확인한 후 선정하도록 한다.
- 성능 테스트 수행 시 성능 목표를 미리 설정하고, 테스트 시나리오를 확정한 다음 테스트를 수행하도록 한다.

## 3-2. 애플리케이션 성능 개선

### 학습 목표

- 코드 최적화 기법, 아키텍처 조정 및 호출 순서 조정 등을 적용하여 애플리케이션 성능을 개선할 수 있다.
- 프로그래밍 언어의 특성에 대한 이해를 기반으로 소스 코드 품질 분석 도구를 활용하여 애플리케이션 성능을 개선할 수 있다.

### 필요 지식 /

#### ① 소스 코드 최적화의 이해

소스 코드 최적화는 읽기 쉽고 변경 및 추가가 쉬운 클린 코드를 작성하는 것으로, 소스 코드 품질을 위해 기본적으로 지킬 원칙과 기준을 정의하고 있다.

##### 1. 나쁜 코드(Bad Code)

###### (1) 개요

다른 개발자가 로직(LogiC)을 이해하기 어렵게 작성된 코드이다. 대표적인 사례로 처리 로직의 제어가 정제되지 않고 서로 얹혀 있는 스파게티 코드, 변수나 메소드에 대한 이름 정의를 알 수 없는 코드, 동일한 처리 로직이 중복되게 작성된 코드 등이 있다.

###### (2) 문제점

스파게티 코드의 경우 찾은 오류가 발생할 가능성이 있다.

소스 코드 이해가 안 되어 계속 덧붙이기할 경우 코드 복잡도가 증가한다.

##### 2. 클린 코드(Clean Code)

###### (1) 개요

잘 작성되어 가독성이 높고, 단순하며, 의존성을 줄이고, 중복을 최소화하여 깔끔하게 잘 정리된 코드를 말한다.

###### (2) 효과성

중복 코드 제거로 애플리케이션의 설계가 개선된다.

가독성이 높으므로 애플리케이션의 기능에 대해 쉽게 이해할 수 있다.

버그를 찾기 쉬워지며, 프로그래밍 속도가 빨라진다.

#### ② 소스 코드 품질 분석 도구의 이해

소스 코드에 대한 코딩 스타일, 설정된 코딩 표준, 코드의 복잡도, 코드 내에 존재하는 메모리 누수 현황, 스레드의 결함 등을 발견하기 위하여 사용하는 분석 도구이며, 정적 분석 도구와 동적 분석 도구가 있다.

## 1. 정적 분석 도구

작성된 소스 코드를 실행시키지 않고, 코드 자체만으로 코딩 표준 준수 여부, 코딩 스타일 적정 여부, 잔존 결함 발견 여부를 확인하는 코드 분석 도구이다.

### (1) 정적 분석 도구의 유형

사전에 결함을 발견하고 예방하는 도구, 코딩 표준 준수 여부를 분석하는 도구, 소스 코드의 복잡도를 계산하는 도구 등이 있다.

## 2. 동적 분석 도구

애플리케이션을 실행하여 코드에 존재하는 메모리 누수 현황을 발견하고, 발생한 스레드의 결함 등을 분석하기 위한 도구이다.

<표 3-4> 소스코드 품질 분석 도구

구분	도구명	도구 설명	지원환경	개발도구 지원	홈페이지
정적 분석 도구	pmd	자바 및 타 언어 소스코드에 대한 버그, 데드코드 분석	Linux, Windows	Eclipse, NetBeans	<a href="https://pmd.github.io">pmd.github.io</a>
	cppcheck	C/C++ 코드에 대한 메모리누수, 오버플로우 등 문제 분석	Windows	Eclipse, gedit	<a href="https://cppcheck.sourceforge.net">cppcheck.sourceforge.net</a>
	SonarQube	소스코드 품질 통합 플랫폼, 플러그인 확장 가능	Cross-Platform	Eclipse	<a href="https://www.sonarqube.org">www.sonarqube.org</a>
	checkstyle	자바 코드에 대한 코딩 표준 준수 검사 도구	Cross-Platform	Ant, Eclipse, NetBeans	<a href="https://checkstyle.sourceforge.net">checkstyle.sourceforge.net</a>
코드 복잡 도	ccm	다양한 언어의 코드 복잡도 분석 도구, CLI 형태 지원	Cross-Platform	Visual Studio	<a href="https://github.com/jonasbluennck/ccm">github.com/jonasbluennck/ccm</a>
	cobertura	jcoverage 기반의 테스트 커버리지 측정 도구	Cross-Platform	Ant, Maven	<a href="https://cobertura.github.io/cobertura">cobertura.github.io/cobertura</a>
	Avalanche	Valgrind 프레임워크 및 STP 기반 소프트웨어 에러 및 취약점 동적 분석 도구	Linux, Android	-	<a href="https://code.google.com/archive/p/avalanche/">code.google.com/archive/p/avalanche/</a>
동적 분석 도구	Valgrind	자동화된 메모리 및 쓰레드 결함 발견 분석 도구	Cross-Platform	Eclipse, NetBeans	<a href="https://valgrind.org">valgrind.org</a>

## 수행 내용 / 애플리케이션 성능 개선하기

---

### 재료 · 자료

- SW 개발 국제 표준(ISO/IEC 9126, ISO/IEC 14598, ISO15504, ISO/IEC 12207, CMMi 등)
- 소프트웨어 테스트 관련 국제 표준(ISO/IEC 29119)
- 요구사항 정의서, 통합 테스트 계획서, 시스템 테스트 계획서, 성능 점검 계획서

### 기기(장비 · 공구)

- 전산 장비(컴퓨터, 프린터, 빔 프로젝터, 네트워크 장비 등)
- OA 도구(워드프로세서, 스프레드시트, 프레젠테이션 도구 등)
- CASE 도구(SW설계 지원 도구, SW 개발 지원 도구, 성능 점검 도구 등)

### 안전 · 유의 사항

- 실험 실습은 정해진 방법과 절차에 따라 실시하여야 한다.
- 테스트 지원 도구의 사용 방법과 유의사항에 대해 습득하여야 한다.
- 테스트 방법별 통합 테스트 시나리오를 작성하여야 한다.
- 테스트 수행은 개발 및 실제 운영 환경과는 분리된 시스템으로 구축하여야 한다.
- 테스팅 서버는 운영 서버와 동일 또는 유사한 성능을 보유한 사양으로 준비하여야 한다.
- 테스트 수행을 위한 데이터는 미리 준비하고 검증하여야 한다.

### 수행 순서

#### ① 애플리케이션 성능 개선 방안을 검토한다.

성능 테스트 결과를 분석하고, 성능 저하 요인별 성능 개선 방안을 계획한다.

##### 1. 성능 저하 요인별 성능 개선 방안을 검토한다.

###### (1) DB 연결 및 쿼리 실행 단계의 성능 개선 방안을 검토한다.

DB Lock, 불필요한 DB Fetch, 네트워크 연결 오류 등에 대해 성능 개선 방안을 검토한다.

###### (2) 메모리 누수, 아키텍처 조정, 호출 순서 변경, 동기화 등 내부 로직 변경으로 인한 성능 개선 방안을 검토한다.

2. 애플리케이션 성능 개선 계획서를 작성한다.

(1) 성능 개선 계획서의 필수 항목을 결정한다.

성능 개선 개요, 성능 개선 수행 전략, 성능 개선 수행 일정 및 절차, 성능 개선 수행 방안에 대해 설정한다.

(2) 선정된 항목을 토대로 성능 개선 계획서를 작성한다.

xx 시스템 제 n 차 확대구축사업 (xx 부문)	성능개선계획서
<u>목차</u>	
1. 성능개선 개요 .....	2
1.1 목적 .....	2
2. 성능개선 수행 전략 .....	4
2.1 대상 구성도 .....	4
2.2 대상 서버 .....	4
2.3 성능개선 목표 .....	5
3. 성능개선 수행 일정 및 절차 .....	5
3.1 수행 일정 .....	5
3.2 수행 절차 .....	6
3.3 개선 절차 .....	6
4. 성능개선 수행 방안 .....	7
4.1 시각화 성능 개선 .....	7
4.2 배치작업 성능 개선 .....	7

[그림 3-5] 성능 개선 계획서 작성 예시

② 코드 최적화 기법을 통한 성능 개선 방안을 작성한다.

나쁜 코드 형식으로 작성된 소스 코드를 클린 코드로 수정하여 성능을 개선할 수 있는 다양한 방안에 대해 검토하고 정리한다.

1. 클린 코드를 작성하기 위한 원칙에 대해 파악한다.

클린 코드의 작성 원칙에 대해 파악하고 아래의 원칙에 대해 정의한다.

(1) 가독성

누구든지 읽기 쉽게 코드를 작성한다.

(2) 단순성

소스 코드는 복잡하지 않고 간단하게 작성한다.

### (3) 의존성

다른 모듈에 미치는 영향을 최소화하도록 작성한다.

### (4) 중복성

중복을 최소화할 수 있는 코드를 작성한다.

### (5) 추상화

상위 클래스/메소드/함수를 통해 애플리케이션의 특성을 간략하게 나타내고, 상세 내용은 하위 클래스/메소드/함수에서 구현한다.

<표 3-5> 클린 코드 작성 원칙

원칙	원칙 설명	비고
기독성	<ul style="list-style-type: none"><li>- 이해하기 쉬운 용어를 사용한다.</li><li>- 코드 작성 시 들여쓰기 기능을 사용한다.</li></ul>	
단순성	<ul style="list-style-type: none"><li>- 한 번에 한 가지 처리만 수행한다.</li><li>- 클래스/메소드/함수를 최소 단위로 분리한다.</li></ul>	
의존성	<ul style="list-style-type: none"><li>- 영향도를 최소화한다.</li><li>- 코드의 변경이 다른 부분에 영향이 없게 작성한다.</li></ul>	
중복성	<ul style="list-style-type: none"><li>- 중복된 코드를 제거한다.</li><li>- 공통된 코드를 사용한다.</li></ul>	
추상화	<ul style="list-style-type: none"><li>- 클래스/메소드/함수에 대해 동일한 수준의 추상화를 한다.</li><li>- 상세 내용은 하위 클래스/메소드/함수에서 구현한다.</li></ul>	

## 2. 소스 코드 최적화 기법의 유형에 대해 파악한다.

클래스/메소드의 분할, 배치, 느슨한 결합(Loosely coupled), 다형성, 코딩 작성 표준 준수, 좋은 이름 사용, 적절한 주석문 사용 등에 대해 파악하고 정리한다.

### (1) 클래스 분할 배치 기법을 파악한다.

클래스는 하나의 역할, 책임만 수행할 수 있도록 응집도를 높이고, 크기를 작게 작성한다.

### (2) 느슨한 결합(Loosely Coupled) 기법을 파악한다.

클래스의 자료 구조, 메소드를 추상화할 수 있는 인터페이스 클래스를 이용하여, 클래스 간의 의존성을 최소화해야 한다.

### (3) 코딩 형식 기법을 파악한다.

줄바꿈으로 개념을 구분, 종속(개념적 유사성 높음) 함수를 사용, 호출하는 함수를 먼저 배치하고 호출되는 함수는 나중에 배치, 변수 선언 위치를 지역 변수는 각 함수 맨 처음에 선언할 때 사용하는 등의 형식을 취한다.

(4) 좋은 이름 사용 방법을 파악한다.

기억하기 좋은 이름, 발음이 쉬운 용어, 접두어 사용 등 기본적인 명명 규칙(Naming Rule)을 정의하고 정의된 이름을 사용한다.

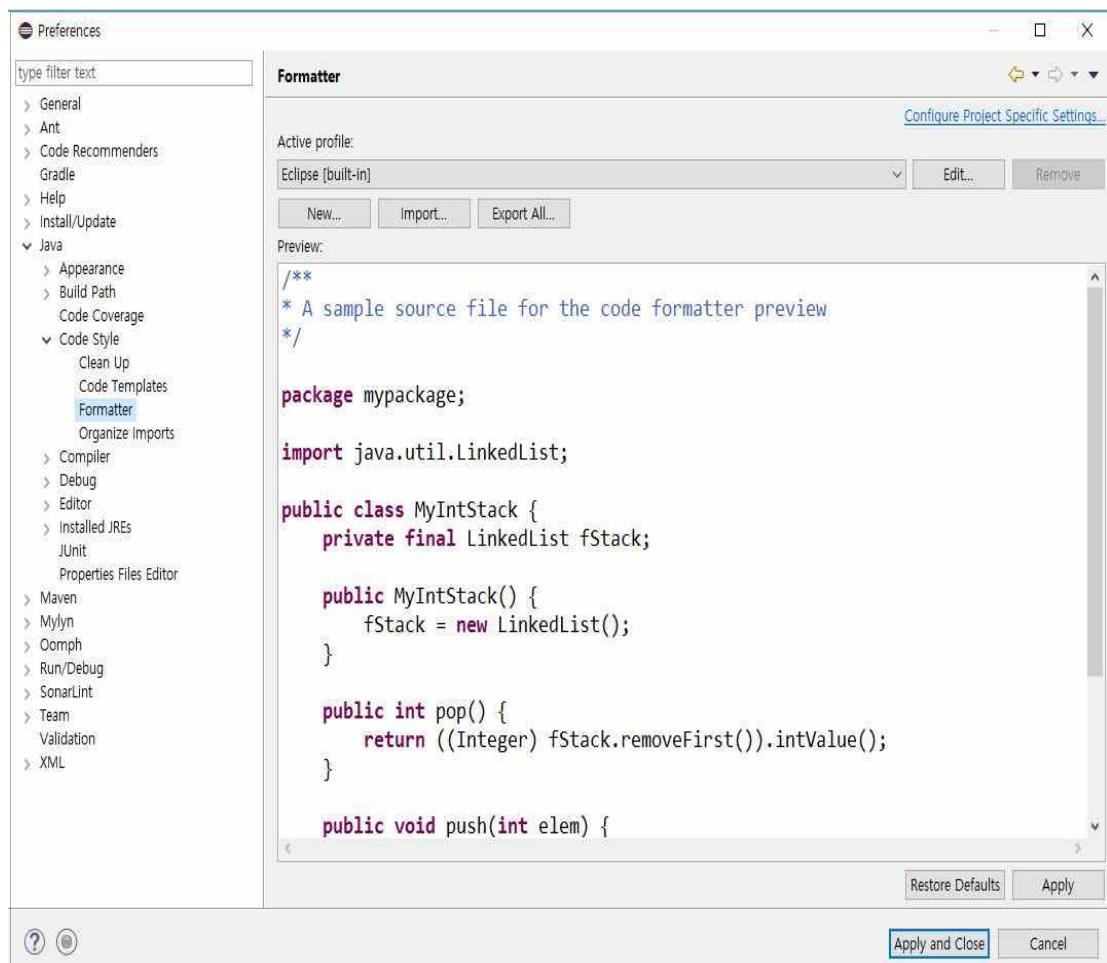
(5) 적절한 주석문 사용 방법을 파악한다.

소스 코드 작업 시 앞으로 해야 할 일을 기록하거나, 소스상의 중요한 부분을 강조할 때 사용한다.

## 2. 소스 코드 최적화 기법을 통해 애플리케이션의 성능을 개선한다.

애플리케이션 개발 프레임워크의 코딩 표준을 설정하고, 인터페이스 클래스를 이용하여 느슨한 결합(Loosely coupled) 코드를 구현한다.

(1) 통합 개발 프레임워크 Eclipse의 JAVA Code Style의 Formatter를 설정한다.



[그림 3-6] Eclipse Code formatter 설정 예시

(2) 인터페이스를 통해 추상화된 자료 구조를 구현하여 의존성을 최소화한다.

```
public class Point {  
    public double x;  
    public double y;  
}  


- 추상화된 자료구조로 의존성 최소화 구현(Loosely coupled)
  - x, y 값이 public 이므로 사용자가 x, y 값을 입력 후 해당 클래스 세부 구현 조작 가능



-> Point 설정 시 항상 x, y 두 값을 한번에 설정해야 함.


```
public interface Point{  
    double getX();  
    double getY();  
    void setRectangular(double x, double y);  
}
```


```

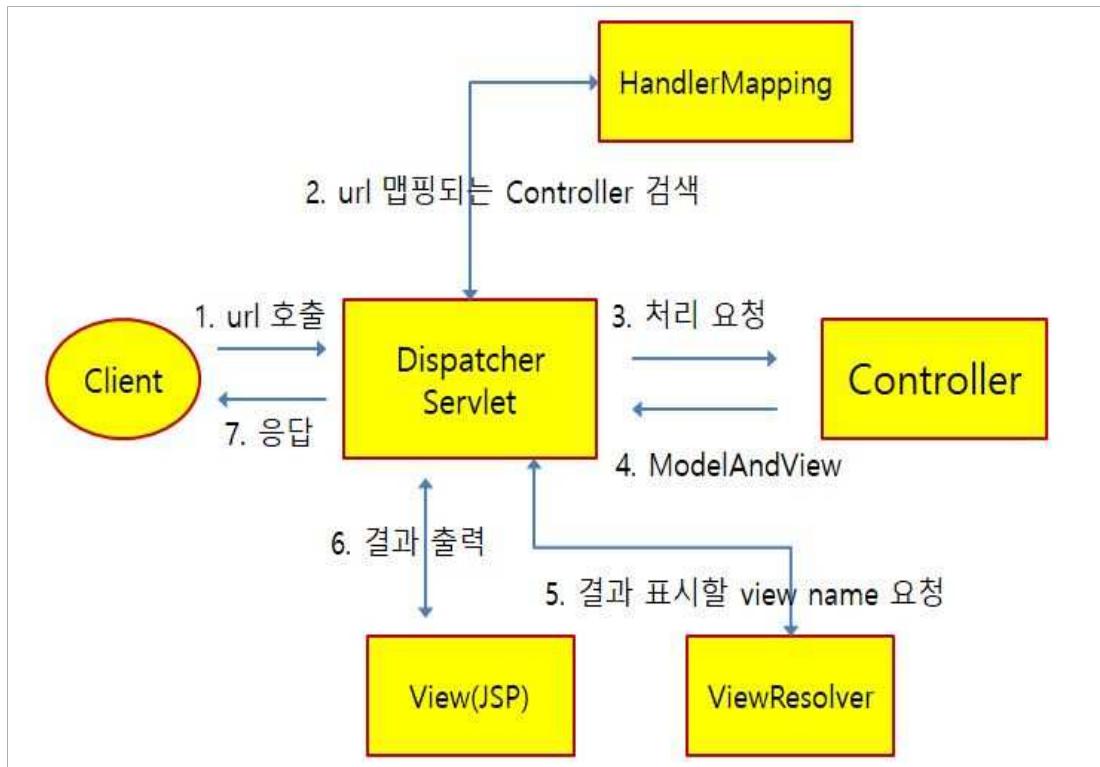
[그림 3-7] 인터페이스 클래스를 이용한 느슨한 결합(Loosely Coupled) 구현 예시

③ 아키텍처 조정을 통한 성능 개선 방안을 작성한다.

소프트웨어 아키텍처 기법 중 사용자 화면 계층(User Interface Layer)의 Spring MVC의 구조 및 각 컴포넌트에 대해 파악하고, 아키텍처 기법인 생성과 사용의 분리 작업을 통해 애플리케이션의 성능을 개선한다.

1. Spring MVC(Model View Controller)의 구조와 각 컴포넌트의 역할에 대해 조사하고 정리한다.

(1) Spring MVC(Model View Controller)의 구조 및 동작 순서에 대해 정리한다.



[그림 3-8] Spring MVC 구조와 동작 순서

(2) Spring MVC의 각 컴포넌트에 대해 정리한다.

(가) DispatcherServlet(디스패처 서블릿)

Spring MVC 프레임워크의 Front Controller, 웹 요청과 응답의 수명주기(Life Cycle)를 주관하는 컴포넌트이다.

(나) HandlerMapping(핸들러매핑)

웹 요청 시 해당 URL에 매핑되는 컨트롤러를 검색, 결정하는 컴포넌트이다.

(다) Controller(컨트롤러)

비즈니스 로직을 수행하고 결과를 ModelAndView에 반영하는 컴포넌트이다.

(라) ModelAndView

수행 결과와 반영하는 모델 데이터 객체, 이동할 페이지 정보 및 뷰로 이루어져 있다.

(마) View Resolver(뷰 리졸버)

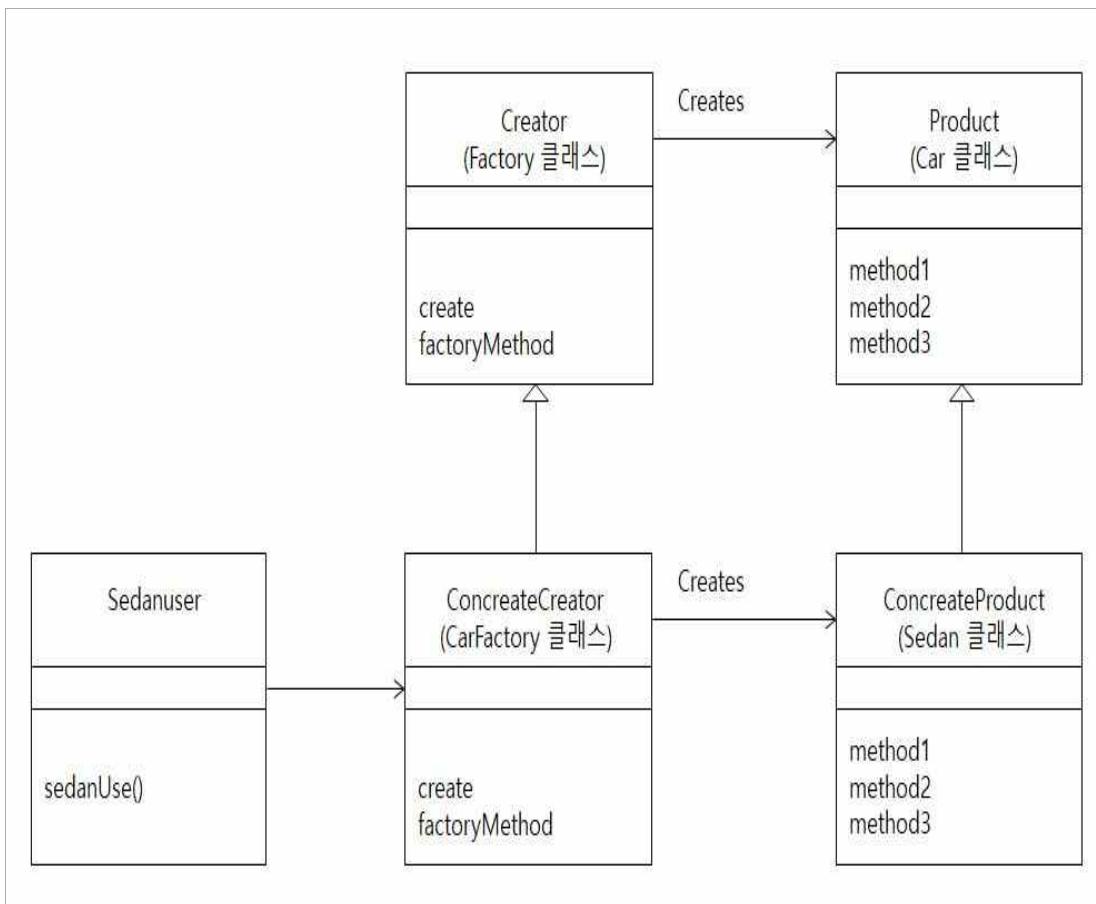
결과를 표시할 어떤 뷰를 선택할지 결정한다.

(바) View

결과 데이터인 모델 객체를 표현(Display)한다.

## 2. 아키텍처 조정을 통한 성능 개선 방안을 수행한다.

객체의 생성과 사용을 분리함으로써 소프트웨어의 의존성을 최소화하기 위하여 팩토리 메소드(Factory Method) 패턴을 이용하여 성능 개선 방안을 수행한다.



[그림 3-9] Factory Method 패턴 사용 예시

## ④ 프로그램 호출 순서 조정을 통한 성능 개선 방안을 작성한다.

서로 연관된 내용은 세로로 가깝게 작성함으로써 밀집도를 높이고, 유사성이 높은 함수나 코드끼리 가깝게 배치한다. 또한 호출하는 함수를 먼저 작성하고, 호출되는 함수는 나중에 배치한다.

### 1. 프로그램 호출 순서를 조정하여 성능 개선 방안을 수행한다.

호출하는 함수를 먼저 코딩하고, 호출되는 함수는 나중에 배치하여 애플리케이션의 성능을 개선한다.

```

public class SeqTest {

    public static void main(String[] args) {

        Seq seq = new Seq();
        System.out.println(seq.cWithdrop(100) );
        System.out.println(seq.cwithdrop(100) );

        int amount = 100;
        if(seq.cWithdrop(amount)==-1)
            Overdrop(); // caller
        else {
            seq.cWithdrop(amount);
            UsualThing(); // caller
        }
    }

    private static void Overdrop() { // callee
        System.out.println("Overdrop() --> ");
    }

    private static void UsualThing() { // callee
        System.out.println("UsualThing() --> ");
    }
}

```

The diagram illustrates the execution flow of the code. It shows two main paths originating from the conditional statement in the main method:

- Path 1 (Red Boxes):** Represented by a red box labeled "Overdrop(); // caller". This path leads to the `Overdrop()` method.
- Path 2 (Red Boxes):** Represented by a red box labeled "UsualThing(); // caller". This path leads to the `UsualThing()` method.
- Callout Labels:** Labels "callee" are placed next to the `Overdrop()` and `UsualThing()` methods, indicating they are called from the "caller" paths above them.

[그림 3-10] 호출 순서 조정을 통한 성능 개선 예시

##### ⑤ 소스 코드 품질 분석 도구를 활용하여 애플리케이션 성능을 개선한다.

###### 1. 메모리 사용 최소화를 통해 성능을 개선한다.

###### (1) String 클래스를 StringBuffer 또는 StringBuilder 클래스로 수정하여 코딩한다.

String 클래스는 연산 시마다 new 객체를 생성하여 Heap 메모리 사용량이 증가함에 따라 속도 저하의 가능성이 있지만, StringBuffer 또는 StringBuilder 클래스가 기존 객체의 크기를 증가시키면서 이를 처리한다.

###### (2) 루프 내 불필요한 메소드의 호출이 반복되지 않도록 코딩한다.

루프 내에서 메소드(함수)의 호출이 반복되는 코드를 루프 진입 전에 호출하도록 소스 코드를 수정하여 성능을 개선한다.

```

public void loopSamp() {
    TrSet trSetSample = null;
    trSetSample = trSetChogi;

    CompareTimer timer = new CompareTimer("loopSamp");
    if(trSetSample != null) {
        int[] treeNum = new int[trSetSample.size()];
        for(int i=0; i<trSetSample.size(); i++) {
            treeNum[i] = (int)trSetSample.toArray()[i];
        }
    }
    timer.checkCurrentTimeMillis();
}

public void loopExec() {
    TrSet trSetSample = null;
    trSetSample = trSetChogi;

    CompareTimer timer = new CompareTimer("loopExec");
    if(trSetSample != null) {
        int[] treeNum = new int[trSetSample.size()];
        int trSetSampleSize = trSetSample.size();
        Iterator<Integer> iterator = trSetSample.iterator();

        for(int i=0; i<trSetSampleSize; i++) {
            treeNum[i] = (int)iterator.next();
        }
    }
    timer.checkCurrentTimeMillis();
}

```

루프 내에서 함수를 계속 호출하는 코드

루프 내에서 함수를 호출하지 않는 코드  
로 성능 개선 가능

[그림 3-11] 루프 호출 최소화 통한 성능 개선 예시

## 2. 입출력 발생 최소화를 통하여 성능을 개선한다.

문자 입력 스트림 또는 파일 정보를 읽어올 때 버퍼(Buffer)를 사용하거나, BufferedReader를 사용하여, 입출력 발생 최소화를 통해 애플리케이션 성능을 개선한다.

```

public class FIOBiGyo {

    public static void main(String[] args) throws Exception {
        String fileName = "C:/frtest2.ppt";

        ArrayList bFIO = FIOBiGyo.rChStream(fileName);
        String bufferFIO = FIOBiGyo.rChStreamBuffer(fileName);
        ArrayList bufferReaderFIO = FIOBiGyo.rBufferReader(fileName);

    }

    public static ArrayList rChStream(String fileName) throws Exception {
        ArrayList<StringBuffer> list = new ArrayList<StringBuffer>();
        FileReader fReader = null;
        CompareTimer timer = new CompareTimer("bFIO");

    }

    public static String rChStreamBuffer(String fileName) throws Exception {
        StringBuffer returnSB = new StringBuffer();
        FileReader fReader = null;
        CompareTimer timer = new CompareTimer("bufferFIO");

    }

    public static ArrayList<String> rBufferReader(String fileName) throws Exception {
        ArrayList<String> returnList = new ArrayList<String>();
        BufferedReader bfReader = null;
        CompareTimer timer = new CompareTimer("bufferReaderFIO");
    }
}

```

[그림 3-12] 입출력 발생 최소화를 통한 성능 개선 예시

### 3. System.out.println()을 사용하지 않음으로써 성능을 개선한다.

파일, 콘솔에 로그를 남기면 애플리케이션 대기 시간이 발생된다. 이에 대응하여 Log4j 로거를 사용함으로써 성능을 개선한다.

## ⑥ 애플리케이션 성능 현황을 관리한다.

### 1. 성능 현황판(Q-Board)을 작성한다.

#### (1) 성능 테스트 수행 단계를 기록한다.

성능 테스트 계획, 성능 테스트 수행 단계, 성능 개선 단계로 나누어 기록한다.

#### (2) 업무 기능별로 할 일, 진행 중, 완료 항목을 준비한다.

수행 단계의 업무 기능별로 앞으로 테스트를 수행할 단위 업무명, 현재 진행 중인 테스트 단위 업무명, 이미 테스트를 완료한 단위 업무명을 작성한다.

(3) 성능 테스트 단계와 각 업무 기능별 매트릭스를 작성한다.

가령 ‘xx응용체계 개발 프로젝트’의 성능 테스트를 진행한다 했을 때, 테스트 대상을 건강/인사관리, OO관리, 교육훈련관리, 종합OOO 등 업무 기능별로 나누고 할 일, 진행 중, 완료 항목에 접착 메모지 등을 이용하여 기록하고 관리한다.

(4) 소멸 차트를 작성한다.

가로축은 날짜를 기록하고, 세로축은 남은 작업량을 기록하여 정해진 날짜가 되었을 때 남은 작업량이 0이 되도록 관리한다.

2. 성능 현황판을 이용하여 애플리케이션 성능을 개선한다.

성능 관리자는 성능 현황판을 이용하여, 성능 테스트 분석 결과 및 성능 저하 원인을 개발자에게 전달함으로써 애플리케이션 성능을 개선하도록 관리한다.



[그림 3-13] 성능 현황판(Q-Board) 작성 예시

**수행 tip**

- 성능 개선 계획을 작성시, 프로그래밍 언어의 특성에 대한 이해가 바탕이 되어 있어야 한다.

## 학습3 교수 · 학습 방법

### 교수 방법

- 애플리케이션의 성능 점검을 위한 방법과 절차에 대한 지식 습득 여부를 확인하고, 수업 을 진행한다.
- 성능 테스트 도구 및 모니터링 도구의 사용 방법과 유의 사항을 설명하고, 학습자의 이해 를 바탕으로 실습을 진행한다.
- 다양한 소스 코드 품질 분석 도구의 활용법에 대해 설명을 하고, 학습자의 이해를 바탕으 로 실습을 지도한다.
- 애플리케이션의 성능을 개선하기 위한 다양한 기법에 대해 설명하고, 이를 통해 성능 개 선 계획서를 작성할 수 있도록 지도한다.
- 성능 개선을 위한 프로그래밍 언어의 특성에 대해 설명하고, 이를 바탕으로 성능 개선 실 습이 가능하도록 지도한다.

### 학습 방법

- 애플리케이션의 성능 점검을 위한 방법과 절차에 대한 지식을 습득하고, 실습을 진행한다.
- 성능 테스트 도구 및 모니터링 도구의 사용 방법과 유의 사항에 대해 학습하고, 이를 바 탕으로 실습을 진행한다.
- 다양한 소스 코드 품질 분석 도구의 활용법에 대해 학습하고, 이해를 바탕으로 실습을 진 행한다.
- 애플리케이션의 성능을 개선하기 위한 다양한 기법에 대해 학습하고, 이를 통해 성능 개 선 계획서를 작성한다.
- 성능 개선을 위한 프로그래밍 언어의 특성에 대해 학습하고, 이를 바탕으로 성능 개선 실 습을 진행한다.

## 학습3 평 가

### 평가 준거

- 평가자는 학습자가 학습 목표를 성공적으로 달성하였는지를 평가해야 한다.
- 평가자는 다음 사항을 평가해야 한다.

학습 내용	학습 목표	성취수준		
		상	중	하
애플리케이션 성능 분석	- 애플리케이션 테스트를 통하여 애플리케이션의 성능을 분석하고, 성능 저하 요인을 발견할 수 있다.			
애플리케이션 성능 개선	- 코드 최적화 기법, 아키텍처 조정 및 호출 순서 조정 등을 적용하여 애플리케이션 성능을 개선할 수 있다. - 프로그래밍 언어의 특성에 대한 이해를 기반으로 소스 코드 품질 분석 도구를 활용하여 애플리케이션 성능을 개선할 수 있다.			

### 평가 방법

- 문제해결 시나리오

학습 내용	평가 항목	성취수준		
		상	중	하
애플리케이션 성능 분석	- 애플리케이션 성능 점검 및 분석에 대한 능력 - 성능 저하 요인 발견 및 개선 계획서 작성 능력			
애플리케이션 성능 개선	- 코딩 표준에 대한 이해 능력 - 코드 최적화에 대한 이해 및 작업 능력 - 소스 코드 품질 분석 도구 활용 능력			

• 평가자 체크리스트

학습 내용	평가 항목	성취수준		
		상	중	하
애플리케이션 성능 분석	- 애플리케이션 성능 점검 방안에 대한 이해 정도			
	- 성능 개선 계획서 작성의 완성도 정도			
애플리케이션 성능 개선	- 프로그래밍 언어별 특징에 따른 코딩 표준 이해 능력			
	- 소스 코드 최적화에 대한 이해 능력			
	- 소스 코드 품질 분석 도구의 특징에 대한 이해 능력			

## 피드백

### 1. 문제해결 시나리오

- 성능 저하 요인을 발견하고 성능 개선 계획서 작성률 평가하여 일정 수준 이하인 학습자는 추가 학습을 진행하고, 재평가하도록 한다.

### 2. 평가자 체크리스트

- 코드 최적화에 대한 이해 능력과 소스 코드 품질 분석 도구의 특징에 대한 이해 능력을 평가한 후 일정 수준 이하인 학습자는 추가 학습을 진행하고, 재평가하도록 한다.

# 참고자료

---



- 공개SW포털. <http://www.oss.kr>.
- 공개SW테스트가이드(2012). 정보통신산업진흥원. <http://www.oss.kr>.
- 국가표준인증 종합정보센터(KSinfo). <http://standard.go.kr>.
- 권원일 · 이현주 · 최승희 · 이승호 · 박은영 · 조현길(개정3판, 2014). 『개발자도 알아야 할 소프트웨어 테스팅 실무』. STA테스팅컨설팅.
- 로버트 C. 마틴, 박재호, 이해영 옮김(2013). 『Clean Code』. 인사이트.
- 스티브 맥코넬, 서우석(2차 개정판, 2017). 『Code Complete』. 위키북스.
- 조대협(2015). 『소프트웨어 개발과 테스트』. 프리렉.
- 최은만(5차 개정판, 2014). 『소프트웨어 공학』. 정의사.
- 한국정보화진흥원. <http://www.nia.or.kr>.

# 활용서식



## 단위 테스트 케이스

테스트 케이스		프로젝트 명 : 대상 시스템 명 :			
단계명 : 단위 테스트		작성자 :	승인자 :	변경 :	
작성일 :		버전 :	테스트 범위 :	테스트 조직 :	
테스트 ID			테스트 일자		
테스트 목적					
테스트 기능					
입력 데이터					
테스트 단계	케이스 설명		예상 출력	중요도	확인
테스트 환경					
전제 조건					
성공/실패 기준					
기타 테스트 의견사항					

## 통합 테스트 시나리오

## 통합 테스트 결과서

결함 관리 대장

## 성능 테스트 케이스

테스트 목표				
목표치	vUser		호출 간격	
	TPS		응답 시간	
성능지표				
시나리오				
사전 확인 사항	테스트 시작 시간		종료 시간	
	스크립트 수행		부하 발생기 상태	
	데이터베이스 상태		투입인력 확인	
	테스트 환경 설정		테스트 데이터	
비고				
최종 평가				
사유				

## 성능 테스트 결과서

## 요구사항 추적 매트릭스(기능)



## NCS학습모듈 개발이력

발행일	2015년 12월 31일	
세분류명	응용SW엔지니어링(20010202)	
개발기관	한국소프트웨어기술진흥협회, 한국직업능력개발원	
집필진	강석진(이비스톰)* 김보운(이화여자대학교) 김홍진(LG CNS) 유은희 장현섭((주)커리텍) 주선태(T3Q) 진권기(이비스톰) 최재준	검토진 김승현(경희대학교) 엄기영(우리에프아이에스) 장온순(한국IT컨설팅) 조상욱(세종대학교) 조성호(삼성카드)
		*표시는 대표집필자임
발행일	2017년 12월 31일	
세분류명	응용SW엔지니어링(20010202)	
개발기관	(사)한국정보통신기술사협회, 한국직업능력개발원	
집필진	박미화(동국대학교)* 김승환((주)캐롯아이) 김원기(LG CNS) 김종명(SM신용정보) 박현기(프리랜서) 유현주((사)한국정보통신기술사협회) 이구성(한국아이씨티(주)) 이숙희(서초문화예술정보학교) 최창선(한빛디엔에스(주)) 홍민표(한화S&C)	검토진 권순명(주씨에이에스) 김태형((사)한국정보통신기술사협회) 양승화(라이나생명보험) 이성화(시스원) 황극인(주)코스콤
		*표시는 대표집필자임
발행일	2018년 12월 31일	
학습모듈명	애플리케이션 테스트 관리(LM2001020226_16v4)	
개발기관	한국직업능력개발원	

## 애플리케이션 테스트 관리(LM2001020226\_16v4)

저작권자	교육부
연구기관	한국직업능력개발원
발행일	2018. 12. 31.

※ 이 학습모듈은 자격기본법 시행령(제8조 국가직무능력표준의 활용)에 의거하여 개발하였으며, NCS통합포털사이트(<http://www.ncs.go.kr>)에서 다운로드 할 수 있습니다.



[www.ncs.go.kr](http://www.ncs.go.kr)