1-1. 테스트 수행
1-2. 결함관리
2-1. 조치 우선순위 결정
2-2. 결함 조치 관리

1-1 테스트 수행
●테스트 개요
테스트 과정에 필요한 역할 : 소프트웨어 아키텍트 / 테스트 매니저

●소프트웨어 생명주기
요구사항-> 분석-> 디자인-> 구현 또는 개발 순으로 진행
프로젝트의 특성과 방법론에 따라 반복적으로 수행하는 경우도 있다.
테스트는 단위테스트-> 통합테스트->시스템테스트->인수테스트 의 순으로 진행

●프로젝트 수행단계에 따른 테스트의 분류
(1) 단위테스트
작은 소프트웨어 단위(컴포넌트 또는 모듈)를 테스트하는 것으로서, 이반적으로 개발자 자신에 의해 행해짐.
(2)통합테스트
모듈 사이의 인터페이스, 통합된 컴포넌트 간의 상호 작용을 테스트 하는 것.
하나의 프로세스가 완성된 경우 부분적으로 통합 테스트를 수행하는 경우도 있다.
(3)시스템 테스트
통합된 단위 시스템의 기능이 시스템에서 정상적으로 수행되는지를 테스트 하는 것
서능 및 장애 테스트가 여기에 포함
(4)인수테스트
일반적으로 최종 사용자와 업무에 딸느 이해관계자 등이 테스트를 수행함.
개발 된 제품에 대해 운영여부를 결정하는 테스트로, 실제 업무 적용 전에 수행

●프로젝트 수행 단계에 따른 테스트의 접근방법
(1) 단위테스트
테스트 가능한 단위로 작게 분리된 소프트웨어 내에서 결함을 찾고 기능을 검증
-특징
구조적 테스트, 기능성 테스트, 리소스 관련 테스트, 강건성 테스트 등 특정 비기능성 테스트 등이 포함되어 수행, 컴포넌트 명세, 소프트웨어 상세설계, 데이터 모델 명세 등을 이용하여 테스트 한다.
-목적
기본적으로 개발된 코드 및 모듈의 범위 내에서 정상적인 작동과 사용자의 요구사항 등을 테스트 하는 데 목적

-방법
ⓐ 구조기반: 업무 단위별 제어흐름과 조건결정에 따른 결과 테스트
프로그램 내부구조 및 복잡도를 검증하는 화이트박스 테스트 / 제어 흐름, 조건결정
ⓑ 명세기반: 동등 분할과 경계 값 분석을 위하여 사용자의 입력, 출력, 내부, 이벤트 등을 확인하는 데 목적
목적 및 실행코드 기반의 실행을 통한 블랙박스테스트/ 동등분할, 경계값분석

(2) 통합테스트
컴포넌트 간 인터페이스 테스트를 하고 운영체제, 파일시스템, 하드웨어 또는 시스템 간 인터페이스와 같은 각각 다른 부분과 상호 연동이 정상적으로 작동하는 지여부 테스트
-특징
일반적으로 빅방 방식 보다는 순차적 형태와 아키텍처에 대한 이해를 바탕
-종류
빅뱅, 상향, 하향, 샌드위치, 센트럴, 콜라보레이션, 레이어 통합 등의 테스트가 있다.

(3) 시스템 테스트
컴퓨터 시스템을 완벽하게 검사하기 위한 목적 또는 성능 목표를 가지고 테스트.
-특징
개발 프로젝트 차원에서 정의된 전체 시스템의 동작과 관련되어 있다
환경 제한적 장애 관련 리스크를 최소화하기 위하여 실제의 최종 사용자 환경과 유사하게 시스템 성능, 관련된 고객의 기능, 비기능적인 요구사항 등이 완벽하게 수행되는지를 테스트하며, 이때 요구사항 명세서, 비즈니스 절차, 유스케이스, 리스크 분석 결과 등을 이용
-방법
ⓐ기능적 요구사항: 요구사항명세서, 비즈니스절차, 유스케이스 등 명세서 기반의 블랙박스 테스트
ⓑ비기능적 요구사항: 성능 테스트, 회복 테스트, 보안 테스트, 내부 시스템의 메뉴 구조, 웹 페이지의 네비게이션 등의 구조적 요소에 대한 화이트박스 테스트

(4)인수테스트
시스템의 일부 또는 특정한 비기능적인 특성에 대해 인수테스트를 통해 확인
-종류
사용자 인수테스트:비즈니스 사용자가 시스템 사용의 적절성 여부를 확인
운영상의 인수테스트:시스템 관리자가 시스템 인수 시 수행하는 테스트 활동으로 백업/복원 시스템, 재난 복구, 사용자 관리, 정기 점검 등을 확인
계약인수테스트:계약상의 인수/검수 조건을 준수하는 지 여부를 확인
규정 인수 테스트:정부지침, 법규, 규정 등 규정에 맞게 개발하였는 지 확인
알파 테스트:개발하는 조직 내 잠재고객에 의해 테스트 수행
베타 테스트:실제 환경에서 고객에 의해 테스트 수행

● 테스트 기반에 따른 테스트 종류
-구조기반: 소프트웨어 내부의 논리 흐름에 따른 테스트 케이스 작성 및 결함 발견 활동
-명세기반: 사용자의 요구사항 분석서에 주어진 명세를 빠뜨리지 않고 테스트 케이스화
-경험기반: 유사 소프트웨어나 기술에서의 테스터의 경험, 직감, 기술 능력을 바탕으로 테스트

●테스트 자동화
-개념
사람이 하던 반복적 테스트 절차를 자동화 도구를 활용하여 준비, 구현, 수행, 분석 등을 스크립트 형태로 구현함으로써, 시간과 인력 투입의 부담을 최소화 하면서 운영중인 시스템의 모니터링 또는 UI가 없는 서비스의 경우에도 정밀한 테스트가 가능
-장점
테스트 데이터의 재입력과 재구성 같은 반복 작업의 자동화를 통하여 테스트 인력과 시간을 최소화
향상된 요구사항 정의, 성능 및 스트레스 테스트, 품질 측정을 최적화
빌드확인, 회귀, 다중 플랫폼 호완성, 소프트웨어 구성, 기본테스트 등의 향상된 테스트 품질 보장
-단점
도입 후 테스트 도구 전문가를 양성 또는 고용 필요 , 초기에 프로세스 적용에 대한 시간, 비용, 노력에 대한 추가 투자가 필요
비공개 상용 소프트웨어의 경우 고가, 인력과 교육에 대한 유지 관리비용 높다.
-고려사항
테스트 절차를 고려하여 재사용 및 측정이 불가능한 테스트 프로그램은 제외해야한다
설계기준 고려하여 반복적인 빌드에서 스크립트 재사용성이 가능해야한다
도구의 한계성으로 모든 수동 테스트 과정을 자동화 할 수 있는 도구는 없다. 따라서 용도에 맞는 적절한 도구 사용이 필요
도구 환경 설정과 도구 습득 기간을 고려하여 프로젝트의 지연을 방지
테스트 엔지니어 늦은 투입은 프로젝트의 이해 부족으로 불완전한 테스틀르 초래할 수 있으믈 적절한 투입 시기와 계획을 프로젝트 초기에 수립해야 한다.

테스트계획-> 요구사항 관리(문서)
테스트 분석/설계->테스트 케이스 생성(문서) / 커버리지 분석
테스트 수행-> 테스트자동화/정적분석/동적분석/성능테스트/모니터링
테스트통제-> 형상관리/테스트관리/결함추적관리

1-2.결함관리
●결함의 정의
프로그램과 명세서 간의 차이, 업무내용 불일치

기대 결과와 실제 관찰 결과 간의 차이
시스템이 사용자가 기대하는 타당한 기대치를 만족시키지 못할 때 변경이 필요한 모든 것

●프로세스
1. 결함관리 계획
2. 결함기록
3. 결함검토
4. 결함수정
5. 결함재확인
6. 결함 상태 추적 및 모니터링 활동
7. 최종 결함 분석 및 보고서 작성

●결함의 상태 및 추적업무 흐름도
결함발견-> 결함등록-> 결함할당-> 결함수정-> 결함종료
결함발견-> 결함등록-> 결함검토-> 결함할당-> 결함수정-> 결함종료
결함발견-> 결함등록-> 결함검토-> 결함해제-> 결함종료
결함발견-> 결함등록-> 결함검토-> 연기, 해결불가->결함조치보류 ->재오픈->결함등록-> 결함할당-> 결함수정-> 결함종료

●결함 분류
1. 시스템 결함
비정상적인 종료/중단, 응답시간 지연, 데이터베이스 에러 등 애플리케이션 환경과 데이터베이스 처리에서 발생하는 결함
2. 기능 결함
사용자의 요구사항 미반영/불일치, 부정확한 비즈니스 프로세스, 스크립트 에러, 타시스템 연동 시 오류 등 기획, 설계, 업무 시나리오 단계에서 발생된 결함
3. GUI결함
응용 프로그램의 UI비일관성, 부정확한 커서와 메시지, 데이터 타입의 표시 오류
4. 문서결함

●결함 심각도
1. High
시스템이 중단 또는 다운되어 더이상 프로세스를 진행할 수 없게 만드는 결함
2. Medium
시스템의 흐름에 영향을 미치는 결함으로 부정확한 기능, 부정확한 업무 프로세스, 데이터 필드 형식의 오류, 데이터베이스 에러, 보안 관련 오류 등

3. Low
시스템의 흐름에는 영향을 미치지 않는 결함이나 상황에 맞지 않는 용도와 화면구성 결함, 부정확한 GUI및 메시지, 에러 시 메시지 미출력, 화면상의 문법/철자 오류 등

●결함에 대한 원인을 분석하고 개선방안 도출
통계적 분석 기법을 적용
결함종류 식별하기 위해 원인분석도구인 파레토 다이어그램과 피시본 다이어그램 사용

2-1. 조치우선순위 결정
●소프트웨어 테스트기법
1. 단위테스트기법
(1) JUnit을 활용한 테스트
(2) Mock 테스트
단위 테스트 시 Mock객체를 사용하여 테스트 하는 기법
테스트 전용 객체를 테스트 더블이라 부른다.
객체유형
ⓐDummy: 객체의 전달에만 사용되고 실제로는 사용하지 않는 것
ⓑFake: 실제로 동작하도록 구현되지만, 실제 환경에서는 다르게 구현할 수 있다.
ⓒStubs: 텟흐트를 위해 미리 준비한 응답만을 제공
ⓓMocks: 스펙을 통해 정의된 응답을 받고, 다른 응답을 받을 경우 예외를 발생하도록 구현, 응답에 대한 확인을 수행하는 역할.

2. 통합테스트기법
전체 시스템이 통합완료될 때까지 단위 시스템 간의 연계성 및 기능 요구사항 들을 확인하고, 하드웨어와 소프트웨어 구성요소 간의 상호 작용을 테스트하는 것이 주요목적.
업무간의 연계성과 상호 운영성 중심의 테스트 수행
-설계기법
개발딘 소프트웨어나 시스템의 요구사항, 요구사항 명세서, 업무구조, 시스템 구조 등을 기반으로 소프트웨어의 어떤 부분을 어떻게 접근하여 테스트 할지에 대한 테스트 상황과 방법 파악
테스트 상황과 방법을 구체화시키기 위한 수단 및 도구를 테스트 설계방법이라 함

●테스트 설계방법
보다 작은 케이스-동등 클래스
보다 많은 버그 찾기-경계테스트
테스트 대상을 빠짐없이 실행하기-모델 기초 테스트 설계

3. 시스템 테스트 기법

(1) 부하 및 성능 테스트

●부하 테스트 기본 공식

동시이용자수

TPS(처리량) * 호출간격(응답시간+대기시간)

동시 단말 이용자

PC앞에서 시스템을 이용하는 이용자 / Active User+(가상적으로)In-Active User

액티브이용자

동시에 같은 서비스나 업무를 실행하고 나서 응답을 기다리고 있는 이용자

처리량

단위 시간당 처리하는 건수

요청을 받는 입장에서 단위시간당 요청건수/단위 시간당 처리건수

대기시간

응답을 받은 직후부터 다음 명령 또는 호출할 때까지 사용자가 대기하는 시간

(2)장애복구테스트

(3)보안테스트

4. 인스테스트 기법

최종사용자가 요구한 기능이 제대로 반영되었는지, 인수조건에 만족하느 ㄴ지를 테스트 하는 기법

●결함관련 용어

1. 에러

소프트웨어 개발 또는 유지 보수 수행 중에 발생한 부정확한 결과

개발자의 실수로 발생한 오타, 개발 명세서의 잘못된 이해, 서브루틴의 기능 오해 등

2. 오류

프로그램 코드상에 존재하는 것으로 비정상적인 프로그램과 정상적인 프로그램 버전 간의 차이로 인하여 발생

잘못된 연산자가 사용된 경우에 프로그램이 서브루틴으로 부터 에러 리턴을 점검하는 코드가 누락된 것

3. 실패

정상적인 프로그램과 비정상적인 프로그램의 실행 결과의 차이를 의미하며, 프로그램 실행 중에 프로그램의 실제 실행 결과를 개발 명세서에 정의도니 예상결과와 비교함으로써 발견

4. 결함

버그, 에러, 오류, 실패, 프로그램 실행에 대한 문제점, 프로그램 개선사항 등의 전체를 포괄하는 용어

●결함 판단 기준

1. 기능 명세서에 가능하다고 명시된 동작이 수행X
2. 기능 명세서에 불가능하다고 명시된 동작이 수행O
3. 기능명세서에 명시되어 있지 않은 동작을 수행하는 경우
4. 기능명세서에 명시되어 있지 않지만 동작을 수행하는 경우
5. 테스터의 시각에서 볼 때 문제가 있다고 판단되는 경우(ex. 이해하기 어령누 기능, 사용이 까다로운 기능, 비정상적으로 느린 기능 등)

2-2.결함 조치 관리

●결함 조치 관리

1. 소프트웨어 인스펙션
2. 인스펙션 중점 항목
3. 코드인스펙션 프로세스
4. 코드인스펙션 태스크와 수행내용
5.설계 인스팩션, (체크리스트)
6. 인스펙션과 워크스루 차이점

●형상관리 및 구성요소

1.소프트웨어 형상 관리의 정의

(1)소프트웨어 프로세스의 모든 출력물 정보

(2)컴퓨터 프로그램, 컴퓨터 프로그램 설명문서, 데이터 등

2. 기준선과 소프트웨어 형상 항목

(1)기준선

(2) 소프트웨어 형상항목

3. 형상관리 주요활동

(1)형상관리기능

(2)형상식별

(3)버전관리-형상관리 구성도