2과목-소프트웨어 개발

(Part 3. 제품 소프트웨어 패키징)

소프트웨어 개발 총 파트

소프트웨어 개발 2과목은 총 5Part로 이루어져있다.

1장 데이터 입, 출력 구현(29.49%)

2장 통합 구현(1.92%)

3장 제품 소프트웨어 패키징(19.87%)

4장 애플리케이션 테스트 관리(35.26%)

5장 인터페이스 구현(13.46%)

제품 소프트웨어 패키징

제품 소프트웨어 패키징 Part는 8개의 섹션으로 구성되어 있다.

01 소프트웨어 패키징

02 릴리즈 노트 작성

03 디지털 저작권 관리(DRM)

04 소프트웨어 설치 매뉴얼 작성

05 소프트웨어 사용자 매뉴얼 작성

06 소프트웨어 버전 등록

07 소프트웨어 버전 관리 도구

08 빌드 자동화 도구

3. 제품 소프트웨어 패키징-SEC_01(소프트웨어 패키징)

- 이 장을 공부하면서 반드시 알아두어야 할 키워드

; 소프트웨어 패키징, 릴리즈 노트, DRM, 소프트웨어 설치 매뉴얼, 소프트웨어 사용자 매뉴얼, 형상 관리, Subversion, Git, Jenkins, Gradle

1) 소프트웨어 패키징의 개요

; 소프트웨어 패키징이란 모듈 별로 생성한 실행 파일들을 묶어 배포용 설치 파일을 만드는 것을 말한다.

- 개발자가 아니라 사용자를 중심으로 진행한다.
- 소스 코드는 향후 관리를 고려하여 모듈화하여 패키징한다.
- 사용자가 소프트웨어를 사용하게 될 환경을 이해하여, 다양한 환경에서 소프트웨어를 손쉽게 사용할 수 있도록 일반적인 배포 형태로 패키징한다.

일반적으로 패키징(Packaging)이란 관련된 것들을 하나로 묶는 것을 말하며, 소프트웨어 패키징이란 기능별로 생성한 실행 파일들을 묶어 배포용 설치 파일을 만드는 것을 의미한다.

모듈화 : 소프트웨어의 성능을 향상시키거나 시스템의 수정 및 재사용, 유지 관리 등이 용이하도록 시스템을 각 기능별로 나누는 것을 말한다.

3. 제품 소프트웨어 패키징-SEC_01(소프트웨어 패키징)

2) 패키징 시 고려사항

- 사용자의 시스템 환경, 즉 운영체제(OS), CPU, 메모리 등에 필요한 최소 환경을 정의한다.
- UI(User Interface)는 사용자가 눈으로 직접 확인할 수 있도록 시각적인 자료와 함께 제공하고 매뉴얼과 일치시켜 패키징한다.
- 소프트웨어는 단순히 패키징하여 배포하는 것으로 끝나는 것이 아니라 하드웨어와 함께 관리될 수 있도록 Managed Service형태로 제공하는 것이 좋다.
- 사용자에게 배포되는 소프트웨어이므로 내부 콘텐츠에 대한 암호화 및 보안을 고려한다.
- 다른 여러 콘텐츠 및 단말기 간 DRM(디지털 저작권 관리) 연동을 고려한다.
- 사용자의 편의성을 위한 복잡성 및 비효율성 문제를 고려한다.
- 제품 소프트웨어 종류에 적합한 암호화 알고리즘을 적용한다.

Managed Service : 고객이 사용 중인 소프트웨어를 24시간 모니터링 하면서 문제 발생 시 현장에 바로 출동하여 필요한 점검을 수행하는 등의 체계적인 운영 관리와 유지 보수를 수행하는 서비스를 의미한다.

3. 제품 소프트웨어 패키징-SEC_01(소프트웨어 패키징)

3) 패키징 작업 순서

; 패키징 주기는 소프트웨어 개발 기법에 따라 달라지는데, 짧은 개발 주기를 반복하는 애자일 기법인 경우에는 보통 2~4주 내에서 지정하며, 각 주기가 끝날 때마다 패키징을 수행한다.

- 프로젝트 개발 과정에서 주기 별로 패키징한 결과물은 테스트 서버에 배포한다.
- 마지막 개발 과정을 거쳐 최종 패키징한 결과물은 고객이 사용할 수 있도록 온라인 또는 오프라인으로 배포한다.
  - 온라인 배포 : 별도로 마련한 운영 서버에 설치 및 사용 매뉴얼과 함께 배포 파일을 등록하여 고객이 직접 다운받아 사용할 수 있도록 한다.
  - 오프라인 배포 : CD-ROM이나 DVD, USB 등에 설치 및 사용 매뉴얼과 함께 배포 파일을 담는다.

최근에는 Eclipse, Visual Studio, Xcode, Android Studio 등의 IDE 도구가 프로그램 코딩부터 배포까지 대부분의 과정을 지원하며, 별도의 버전 관리 프로그램을 연동하면 버전 관리 작업까지도 지원하기 때문에 별도의 패키징 도구를 사용하지 않는다. 패키징 된 소프트웨어는 사용자가 직접 다운받아 설치할 수 있도록 웹 사이트나 앱 스토어 등에 등록되어 배포된다.

3. 제품 소프트웨어 패키징-SEC_01(소프트웨어 패키징)

3) 패키징 작업 순서

작성된 코드의 기능을 확인한다.

확인된 기능 단위로 코드들을 분류한다.

모듈 단위별로 실행 파일을 만든다.

웹, 모바일, PC 등 소프트웨어가 사용될 환경이나 운영체제, CPU, RAM 등의 최소 운영 환경을 정의한다.

•빌드된 실행 파일들을 정의된 환경에 맞게 배포용 파일 형식으로 패키징 한다.
•정의된 환경과 동일한 환경에서 패키징 결과를 테스팅한 후 소프트웨어에 대한 불편사항을사용자 입장에서 확인한다.

확인된 불편 사항을 반영하기 위한 패키징의 변경 및 개선을 진행한다.

배포 수행 시 오류가 발생하면 해당 개발자에게 전달하여 수정을 요청한다.

빌드(Build) : 빌드는 소스 코드 파일들을 컴퓨터에서 실행할 수 있는 제품 소프트웨어로 변환하는 과정 또는 결과물을 말한다.

사용자 환경 분석 : 사용자 실행 환경은 운영체제(OS), 시스템 사양, 사용 방법 등 을 최대한 자세하게 구분하여 미리 정의해 놓아야 하며, 실행 환경이 다양한 경우에는 각 환경 별로 배포본을 만들기 위해 여러 번의 패키징을 수행해야 한다. 예를 들면, Windows 10 운영체제는 32비트와 64비트를 선택하여 설치할 수 있도록 Win10_32bit.msi와 Win10_64bit.msi 두 가지 버전으로 패키징하여 배포된다.

주요 배포용 파일 형식

•msi : Windows용 패키지 형식
•dmg : Mac OS용 패키지 형식
•jar : java 응용 소프트웨어나 라이브러리를 배포하기 위한 패키지 형식
•war : java Servlet, java Class, xml 및 웹 애플리케이션 서비스를 제공하기 위한 패키지 형식
•ear : jar와 war를 묶어 하나의 애플리케이션 서비스를 제공할 수 있는 패키지 형식
•apk : 안드로이드용 앱 패키지 형식
•ipa: iOS용 앱 패키지 형식

3. 제품 소프트웨어 패키징-SEC_01(소프트웨어 패키징) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(소프트웨어 패키징)

1. 소프트웨어 패키징 도구 활용 시 고려사항으로 틀린 것은?

① 반드시 내부 콘텐츠에 대한 암호화 및 보안을 고려한다.

② 보안을 위하여 이기종 연동을 고려하지 않아도 된다.

③ 사용자 편의성을 위한 복잡성 및 비효율성 문제를 고려한다.

④ 제품 소프트웨어 종류에 적합한 암호화 알고리즘을 적용한다.

이기종 연동이라는 하나 이상의 프로세서 또는 코어를 사용하는 시스템을 가리킨다.

패키징 도구를 활용하여 할 때는 다른 여러 컨텐츠 및 단말기 간 연동을 반드시 고려해야 한다.

사용자의 시스템 환경, 즉 운영체제(OS), CPU, 메모리, 그래픽 카드 등에 필요한 최소 환경을 정의해야 한다.

UI는 사용자가 눈으로 직접 확인할 수 있도록 시각적인 자료와 함께 제공하고 매뉴얼과 일치시켜 패키징을 해야 한다.

소프트웨어는 단순히 패키징하여 배포하는 것으로 끝나는 것이 아니라 하드웨어와 함께 관리될 수 있도록 Managed Service형태로 제공하는 것이 좋다. 사용자에게 배포되는 소프트웨어이므로 내부 컨텐츠에 대한 암호화 및 보안을 고려해야 한다.

3. SW 패키징 도구 활용 시 고려사항과 거리가 먼 것은?

① 패키징 시 사용자에게 배포되는 SW이므로 보안을 고려한다.

② 사용자 편의성을 위한 복잡성 및 비효율성 문제를 고려한다.

③ 보안상 단일 기종에서만 사용할 수 있도록 해야 한다.

④ 제품 SW종류에 적합한 암호화 알고리즘을 적용한다.

4. 다음 중 패키징 작업 과정에 대한 설명으로 잘못된 것은?

① 짧은 개발 주기를 반복하는 애자일 기법인 경우 패키징 주기는 보통 2~4주 내에서 지정하며, 모든 주기가 완료된 후에 최종적으로 패키징을 수행한다.

② 패키징한 결과물을 온라인으로 배포할 때는 별도로 마련한 운영 서버에 설치 및 사용 매뉴얼과 함께 배포 파일을 등록하여 고객이 직접 다운받아 사용할 수 있도록 한다.

③ 패키징한 결과물을 오프라인으로 배포할 때는 CD-ROM이나 DVD, USB 등에 설치 및 사용 매뉴얼과 함께 배포 파일을 담는다.

④ 프로젝트 개발 과정에서 패키징한 결과물은 테스트 서버에 배포한다.

패키징 주기는 소프트웨어 개발 기법에 따라 달라지는데, 짧은 개발 주기를 반복하는 애자일 기법인 경우에는 보통 2~4주 내에서 지정하며,

3. 제품 소프트웨어 패키징-SEC_01(소프트웨어 패키징) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(소프트웨어 패키징)

5. 다음 중 패키징 과정에서 수행하는 작업에 대한 설명으로 잘못 된 것은?

① 모듈화 : 모듈 단위별로 실행 파일을 만든다.

② 기능 식별 : 작성된 코드의 기능을 확인한다.

③ 적용 시험 : 정의된 환경과 동일한 환경에서 패키징 결과를 테스팅 한다.

④ 배포 : 배포 수행 시 오류가 발생하면 해당 개발자에게 전달 하여 수정을 요청한다.

패키징 작업 순서

기능 식별 -> 모듈화(확인된 기능 단위로 코드들을 분류한다) -> 빌드 진행(모듈 단위별로 실행파일을 만든다) -> 사용자 환경분석 -> 패키징 및 적용시험 -> 패키징 변경 개선 -> 배포 순으로 이루어진다.

6. 다음 중, 주요 배포용 파일 형식이 잘못된 것은?

① dmg : jar와 war를 묶어 하나의 애플리케이션 서비스를 제공할 수 있는 패키지 형식

② msi: Windows용 패키지 형식

3. 제품 소프트웨어 패키징-SEC_02(릴리즈 노트 작성)

1) 릴리즈 노트(Release Note)의 개요

; 릴리즈 노트는 개발 과정에서 정리된 릴리즈 정보를 소프트웨어의 최종 사용자인 고객과 공유하기 위한 문서이다.

- 릴리즈 노트를 통해 테스트 진행 방법에 대한 결과와 소프트웨어 사양에 대한 개발팀의 정확한 준수 여부를 확인할 수 있다.
- 소프트웨어에 포함된 전체 기능, 서비스의 내용, 개선 사항 등을 사용자와 공유할 수 있다.
- 릴리즈 노트를 이용해 소프트웨어의 버전 관리나 릴리즈 정보를 체계적으로 관리할 수 있다.
- 릴리즈 노트는 소프트웨어의 초기 배포 시 또는 출시 후 개선 사항을 적용한 추가 배포 시에 제공한다.
- 소프트웨어의 초기 배포 시 제공되는 릴리즈 노트에서는 소프트웨어에 포함된 기능이나 사용 환경에 대한 내용을 확인할 수 있다.
- 소프트웨어 출시 후 개선된 작업이 있을 때마다 관련 내용을 릴리즈 노트에 담아 제공한다.
- 릴리즈 노트에 정리된 정보들은 철저한 테스트를 거친 것이며, 개발팀에서 제공하는 소프트웨어 사양에 대한 최종 승인을 얻은 후 문서화 되어 제공된다.

3. 제품 소프트웨어 패키징-SEC_02(릴리즈 노트 작성)

2) 릴리즈 노트 초기 버전 작성 시 고려사항

; 릴리즈 노트의 초기 버전은 다음의 사항을 고려하여 작성한다.

- 릴리즈 노트는 정확하고 완전한 정보를 기반으로 개발팀에서 직접 현재 시제로 작성해야 한다.
- 신규 소스, 빌드 등의 이력이 정확하게 관리되어 변경 또는 개선된 항목에 대한 이력 정보들도 작성 되어야 한다.
- 릴리즈 노트 작성에 대한 표준 형식은 없지만 일반적으로 다음과 같은 항목이 포함된다.

Header(머릿말)	릴리즈 노트 이름, 소프트웨어 이름, 릴리즈 버전, 릴리즈 날짜, 릴리즈 노트 날짜, 릴리즈 노트 버전 등
개요	소프트웨어 및 변경사항 전체에 대한 간략한 내용
목적	해당 릴리즈 버전에서의 새로운 기능이나 수정된 기능의 목록·릴리즈 노트의 목적에 대한 간략한 개요
문제 요약	수정된 버그에 대한 간략한 설명 또는 릴리즈 추가 항목에 대한 요약
재현 항목	버그 발견에 대한 과정 설명
수정/개선 내용	버그를 수정/개선한 내용을 간단히 설명
사용자 영향도	사용자가 다른 기능들을 사용하는데 있어 해당 릴리즈 버전에서의 기능 변화가 미칠 수 있는 영향에 대한 설명
SW 지원 영향도	해당 릴리즈 버전에서의 기능 변화가 다른 응용 프로그램들을 지원하는 프로세스에 미칠 수 있는 영향에 대한 설명
노트	SW/HW 설치 항목, 업그레이드, 소프트웨어 문서화에 대한 참고 항목
면책 조항	회사 및 소프트웨어와 관련하여 참조할 사항 예) 프리웨어, 불법 복제 금지 등
연락처	사용자 지원 및 문의 응대를 위한 연락처 정보

3. 제품 소프트웨어 패키징-SEC_02(릴리즈 노트 작성)

3) 릴리즈 노트 추가 버전 작성 시 고려사항

; 소프트웨어의 테스트 과정에서 베타 버전이 출시되거나 긴급한 버그 수정, 업그레이드와 같은 자체 기능 향상, 사용자 요청 등의 특수한 상황이 발생하는 경우 릴리즈 노트를 추가로 작성한다.

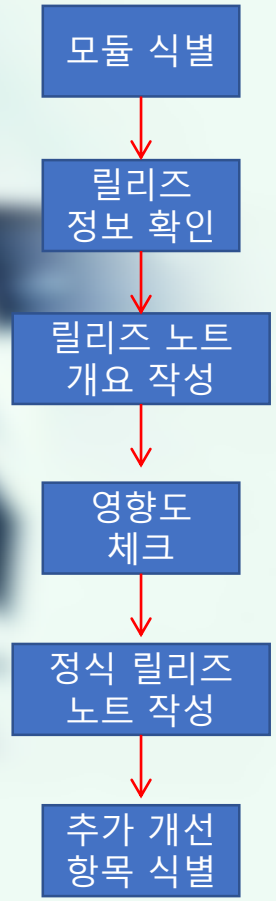
- 중대한 오류가 발생하여 긴급하게 수정하는 경우에는 릴리즈 버전을 출시하고 버그 번호를 포함한 모든 수정된 내용을 담아 릴리즈 노트를 작성한다.
- 소프트웨어에 대한 기능 업그레이드를 완료한 경우에는 릴리즈 버전을 출시하고 릴리즈 노트를 작성한다.
- 사용자로부터 접수된 요구사항에 의해 추가나 수정된 경우 자체 기능 향상과는 다른 별도의 릴리즈 버전으로 출시하고 릴리즈 노트를 작성한다.

베타 버전(Beta Version) : 베타 버전은 소프트웨어를 정식으로 출시하기 전에 테스트할 목적으로 지정된 일부 사용자들 에게만 배포하는 시험용 소프트웨어이다.

3. 제품 소프트웨어 패키징-SEC_02(릴리즈 노트 작성)

4) 릴리즈 노트 작성 순서

; 릴리즈 노트는 일반적으로 다음과 같은 순서로 작성한다.



모듈별 빌드 수행 후 릴리즈 노트에 작성될 내용들을 확인한다.

릴리즈 노트 이름, 소프트웨어 이름, 릴리즈 버전, 릴리즈 날짜, 노트 날짜, 노트 버전 등을 확인한다.

소프트웨어 및 변경사항 전체에 대한 간략한 내용을 작성한다.

버그나 이슈 관련 내용 또는 해당 릴리즈 버전에서의 기능 변화가 다른 소프트웨어나 기능을 사용하는데 미칠 수 있는 영향에 대해 기술한다.

Header(머릿말), 개요, 영향도 체크 항목을 포함하여 정식 릴리즈 노트에 작성될 기본 사항들을 작성한다.

추가 버전 릴리즈 노트 작성이 필요한 경우 추가 릴리즈 노트를 작성한다.

3. 제품 소프트웨어 패키징-SEC_02(릴리즈 노트 작성) 출제 예상 문제

출제 예상 문제(릴리즈 노트 작성)

1. 다음이 설명하는 것은 무엇인가?

개발 과정에서 소프트웨어가 얼마나 개선되었는지를 정리한 정보를 사용자와 공유하기 위해 작성하는 문서로, 이를 통해 사용자는 소프트웨어에 포함된 서비스나 사용 환경 등을 확인할 수 있다.

① 요구사항 명세서(Requirement Specification)

② 릴리즈 노트(Release Note)

③ 소프트웨어 매뉴얼(Software Manual)

④ 소프트웨어 개발 계획서(Software Development Plan)

릴리즈 노트는 개발 과정에서 정리된 릴리즈 정보를 소프트웨어의 최종 사용자인 고객과 공유하기 위한 문서이다. 릴리즈 노트를 통해 테스트 진행 방법에 대한 결과와 소프트웨어 사양에 대한 개발팀의 정확한 준수 여부를 확인할 수 있다. 소프트웨어에 포함된 전체 기능, 서비스의 내용, 개선 사항 등을 사용자와 공유할 수 있다. 릴리즈 노트를 이용해 소프트웨어의 버전 관리나 릴리즈 정보를 체계적으로 관리할 수 있다.

릴리즈 노트는 소프트웨어의 초기 배포 시 또는 출시 후 개선 사항을

3. 다음 중 릴리즈 노트에 대한 설명으로 잘못된 것은?

① 릴리즈 노트는 개발팀에서 제공하는 사양에 대한 최종 승인까지 얻은 후 문서화 되어 사용자에게 제공한다.

② 릴리즈 노트는 정확하고 완전한 정보를 기반으로 개발팀에서 직접 현재 시제로 작성해야 한다.

③ 중대한 오류가 발생하여 이를 긴급하게 수정하는 경우에는 별도로 패키징을 수행해서 재배포를 수행하므로 이와 관련된 릴리즈 노트는 작성하지 않아도 된다.

④ 자체적으로 소프트웨어에 대한 기능 업그레이드를 완료한 경우 정식으로 릴리즈 버전을 추가하고 이에 따른 릴리즈 노트를 작성한다.

소프트웨어의 테스트 과정에서 베타 버전이 출시되거나 긴급한 버그 수정, 업그레이드와 같은 자체 기능 향상, 사용자 요청 등의 특수한 상황이 발생하는 경우 릴리즈 노트를 추가적으로 작성해야 한다.

4. 다음 중 릴리즈 노트의 작성에서 '버그나 이슈 관련 내용 또는 해당 릴리즈 버전에서의 기능 변화가 다른 소프트웨어나 기능을 사용 하는데 미칠 수 있는 영향에 대해 기술한다.'는 어떤 부분을 말하는가?

① 모듈 식별

② 영향도 체크

3. 제품 소프트웨어 패키징-SEC_02(릴리즈 노트 작성) 출제 예상 문제

출제 예상 문제(릴리즈 노트 작성)

5. 릴리즈 노트 작성에 대한 표준 형식은 없지만 일반적으로 다음 중에서 포함되지 말아야 할 항목은 무엇인가?

① 머리말　　② 코드 소스

③ 개요　　④ 면책 조항

릴리즈 노트 작성에 대한 표준 형식은 없지만 일반적으로 다음과 같은 항목이 포함된다.

머리말, 개요, 목적, 문제 요약, 재현 항목, 수정/개선 내용, 사용자 영향도, SW지원 영향도, 노트, 면책 조항, 연락처 등을 포함한다.

3. 제품 소프트웨어 패키징-SEC_03(디지털 저작권 관리(DRM))

1) 저작권의 개요

; 저작권이란 소설, 시, 논문, 강연, 연술, 음악, 연극, 무용, 회화, 서예, 건축물, 사진, 영상, 지도, 도표, 컴퓨터 프로그램 저작물 등에 대하여 창작자가 가지는 배타적 독점적 권리로 타인의 침해를 받지 않을 고유한 권한이다.

● 컴퓨터 프로그램들과 같이 복제하기 쉬운 저작물에 대해 불법 복제 및 배포 등을 막기 위한 기술적인 방법을 통칭해 **저작권 보호 기술**이라고 한다.

3. 제품 소프트웨어 패키징-SEC_03(디지털 저작권 관리(DRM))

2) 디지털 저작권 관리(DRM; Digital Right Management)의 개요

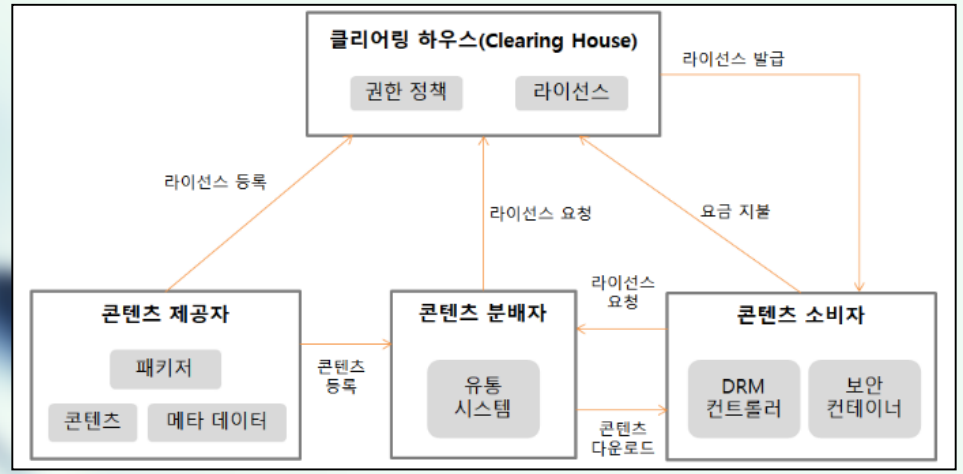
; 디지털 저작권 관리는 저작권자가 배포한 디지털 콘텐츠가 저작권자가 의도한 용도로만 사용되도록 디지털 콘텐츠의 생성, 유통, 이용까지의 전 과정에 걸쳐 사용되는 디지털 콘텐츠 관리 및 보호 기술이다.

- 원본 콘텐츠가 아날로그인 경우에는 디지털로 변환한 후 패키저(Packager)에 의해 DRM 패키징을 수행한다.
- 콘텐츠의 크기에 따라 음원이나 문서와 같이 크기가 작은 경우에는 사용자가 콘텐츠를 요청하는 시점에서 실시간으로 패키징을 수행하고, 크기가 큰 경우에는 미리 패키징을 수행한 후 배포한다.
- 패키징을 수행하면 콘텐츠에는 암호화된 저작권자의 전자서명이 포함되고 저작권자가 설정한 라이선스 정보가 클리어링 하우스(Clearing House)에 등록된다.
- 사용자가 콘텐츠를 사용하기 위해서는 클리어링 하우스에 등록된 라이선스 정보를 통해 사용자 인증과 콘텐츠 사용 권한 소유 여부를 확인 받아야 한다.
- 종량제 방식을 적용한 소프트웨어의 경우 클리어링 하우스를 통해 서비스의 실제 사용량을 측정하여 이용한 만큼의 요금을 부과한다.

클리어링 하우스(Clearing House) : 클리어링 하우스는 디지털 저작권 라이선스의 중개 및 발급을 수행하는 곳으로 디지털 저작물의 이용 내역을 근거로 저작권료의 정산 및 분배가 수행된다.
종량제 방식 : 실제 사용한 양에 따라 요금을 차등 적용하는 방식을 말한다.

3. 제품 소프트웨어 패키징-SEC_03(디지털 저작권 관리(DRM))

3) 디지털 저작권 관리의 흐름 및 구성 요소



- **클리어링 하우스(Clearing House)** : 저작권에 대한 사용 권한, 라이선스 발급, 암호화된 키 관리, 사용량에 따른 결제 관리 등을 수행하는 곳
- **콘텐츠 제공자(Contents Provider)** : 콘텐츠를 제공하는 저작권자
- **패키저(Packager)** : 콘텐츠를 메타 데이터와 함께 배포 가능한 형태로 묶어 암호화하는 프로그램
- **콘텐츠 분배자(Contents Distributor)** : 암호화된 콘텐츠를 유통하는 곳이나 사람
- **콘텐츠 소비자(Customer)** : 콘텐츠를 구매해서 사용하는 주체
- **DRM 컨트롤러(DRM Controller)** : 배포된 콘텐츠의 이용 권한을 통제하는 프로그램
- **보안 컨테이너(Security Container)** : 콘텐츠 원본을 안전하게 유통하기 위한 전자적 보안 장치

3. 제품 소프트웨어 패키징-SEC_03(디지털 저작권 관리(DRM))

4) 디지털 저작권 관리의 기술 요소

; 디지털 저작권 관리를 위해 사용되는 기술은 다음과 같다.

구성 요소	설명
암호화(Encryption)	콘텐츠 및 라이선스를 암호화하고 전자 서명을 할 수 있는 기술
키 관리(Key Management)	콘텐츠를 암호화한 키에 대한 저장 및 분배 기술
암호화 파일 생성(Packager)	콘텐츠를 암호화된 콘텐츠로 생성하기 위한 기술
식별 기술(Identification)	콘텐츠에 대한 식별 체계 표현 기술
저작권 표현(Right Expression)	라이선스의 내용 표현 기술
정책 관리(Policy Management)	라이선스 발급 및 사용에 대한 정책 표현 및 관리 기술
크랙 방지(Tamper Resistance)	크랙에 의한 콘텐츠 사용 방지 기술
인증(Authentication)	라이선스 발급 및 사용의 기준이 되는 사용자 인증 기술

전자 서명(Digital Signature) : 전자 서명이란 전자 문서의 변경 여부를 확인할 수 있도록 작성자의 고유 정보를 암호화하여 문서에 포함 하는 기술을 의미한다.

크랙(Crack) : 크랙이란 '깨다', '부수다'라는 의미 그대로 불법적인 방법으로 소프트웨어에 적용된 저작권 보호 기술을 해제하여 무단으로 사용할 수 있도록 하는 기술이나 도구를 말한다.

3. 제품 소프트웨어 패키징-SEC_03(디지털 저작권 관리(DRM)) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(디지털 저작권 관리(DRM))

1. 저작권 관리 구성 요소에 대한 설명이 틀린 것은?

① 콘텐츠 제공자(Contents Provider) : 콘텐츠를 제공하는 저작권자

② 콘텐츠 분배자(Contents Distributor) : 콘텐츠를 메타데이터 와 함께 배포 가능한 단위로 묶는 기능

③ 클리어링 하우스(Clearing House) : 키 관리 및 라이선스 발급 관리

④ DRM 컨트롤러 : 배포된 콘텐츠의 이용 권한을 통제

클리어링 하우스(Clearing House) : 저작권에 따른 사용 권한, 라이선스 발급, 암호화된 키 관리, 사용량에 따른 결제 관리 등을 수행하는 곳

콘텐츠 제공자(Content Provider) : 콘텐츠를 제공하는 저작권자

패키저(Packager) : 콘텐츠를 메타 데이터와 함께 배포 가능한 형태로 묶어 암호화 하는 프로그램

콘텐츠 분배자(Contents Distributor) : 암호화된 콘텐츠를 유통하는 곳이나 사람

콘텐츠 소비자(Customer) : 콘텐츠를 구매해서 사용하는 주체

DRM 컨트롤러(DRM Controller) : 배포된 콘텐츠의 이용 권한을 통제하는 프로그램

3. 디지털 저작권 관리(DRM)의 기술 요소가 아닌 것은?

① 크랙 방지 기술 ② 정책 관리 기술

③ 암호화 기술 ④ 방화벽 기술

방화벽 기술은 기업이나 조직 내부에 네트워크와 인터넷 간에 전송되는 정보를 선별하여 수용, 거부, 수정하는 침입 차단 시스템이다.

4. 디지털 저작권 관리(DRM) 구성 요소가 아닌 것은?

① Dataware House ② DRM Controller

③ Packager ④ Contents Distributor

Dataware House가 아니라 Clearing House가 디지털 저작권 관리의 구성 요소이다.

3. 제품 소프트웨어 패키징-SEC_03(디지털 저작권 관리(DRM)) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(디지털 저작권 관리(DRM))

5. DRM(Digital Rights Management)과 관련한 설명으로 틀린 것은?

① 디지털 콘텐츠와 디바이스의 사용을 제한하기 위해 하드웨어 제조업자, 저작권자, 출판업자 등이 사용할 수 있는 접근 제어 기술을 의미한다.

② 디지털 미디어의 생명 주기 동안 발생하는 사용 권한관리, 과금, 유통 단계를 관리하는 기술로도 볼 수 있다.

③ 클리어링 하우스(Clearing House)는 사용자에게 콘텐츠 라이선스를 발급하고 권한을 부여해주는 시스템을 말한다.

④ 원본을 안전하게 유통하기 위한 전자적 보안은 고려하지 않기 때문에 불법 유통과 복제의 방지는 불가능하다.

보안 컨테이너(Security Container) : 콘텐츠 원본을 안전하게 유통하기 위한 전자적 보안 장치가 있다.

6. 다음이 설명하는 것은 무엇인가?

소설, 시, 논문, 강연, 연술, 음악, 연극, 무용, 회화, 서예 ,건축물, 사진, 영상, 지도, 도표, 컴퓨터 프로그램 저작물 등에 대하여 창작자가 가지는 배타적 독점적 권리로 타인의 침해를 받지 않을 고유한 권한이다.

7. 다음 중, 디지털 저작권 관리를 위해 사용되는 기술 중 '콘텐츠 및 라이선스를 암호화하고 전자 서명을 할 수 있는 기술'에 해당하는 것은?

① 정책 관리 ② 정책 관리 기술

③ 암호화 기술 ④ 식별 기술

콘텐츠 및 라이선스를 암호화하고 전자 서명을 할 수 있는 기술은 암호화 기술이다.

8. 다음 중, 불법적인 방법으로 소프트웨어에 적용된 저작권 보호 기술을 해제하여 무단으로 사용할 수 있도록 하는 기술이나 도구를 무엇 이라 하는가?

① 크랙 ② 전자 서명

③ 암호화 ④ 인증

크랙이란 '깨다', '부수다'라는 의미 그대로 불법적인 방법으로 소프트웨어에 적용된 저작권 보호 기술을 해제하여 무단으로 사용할 수 있도록 하는 기술이나 도구를 말한다.

3. 제품 소프트웨어 패키징-SEC_04(소프트웨어 설치 매뉴얼 작성)

1) 소프트웨어 설치 매뉴얼 작성의 개요

; 소프트웨어 설치 매뉴얼은 개발 초기에서부터 적용된 기준이나 사용자가 소프트웨어를 설치하는 과정에 필요한 내용을 기록한 설명서와 안내서이다.

- 설치 매뉴얼은 사용자 기준으로 작성한다.
- 설치 시작부터 완료할 때까지의 전 과정을 빠짐없이 순서대로 설명한다.
- 설치 과정에서 표시될 수 있는 오류 메시지 및 예외 상황에 관한 내용을 별도로 분류하여 설명한다.
- 소프트웨어 설치 매뉴얼에는 목차 및 개요, 서문, 기본 사항 등이 기본적으로 포함되어야 한다.
- 소프트웨어 설치 매뉴얼의 목차에는 전체 설치 과정을 순서대로 요약한 후 관련 내용의 시작 페이지를 함께 기술한다.
- 소프트웨어 설치 매뉴얼의 개요에는 설치 매뉴얼의 주요 특징, 구성과 설치 방법, 순서 등의 내용을 기술한다.

3. 제품 소프트웨어 패키징-SEC_04(소프트웨어 설치 매뉴얼 작성)

2) 서문

; 서문에는 문서 이력, 설치 매뉴얼의 주석, 설치 도구의 구성, 설치 환경 체크 항목을 기술한다.

● 문서 이력

버전	작성자	작성일	검토자	일시	검수인
v0.1	신은혁	2014-05-20	황태자	2014-05-25	박인식
변경 내용	최초 작성				
v0.1	신은혁	2014-06-20	황태자	2014-06-25	박인식
변경 내용	주의 사항과 참고 사항을 기술한다. 로고 변경				

● 설치 매뉴얼의 주석

- 주의 사항 : 소프트웨어를 설치할 때 사용자가 반드시 알고 있어야 하는 중요한 내용을 기술한다.
- 참고 사항 : 설치에 영향을 미칠 수 있는 사용자의 환경이나 상황에 대한 내용을 기술한다.

● 설치 도구의 구성

- exe, dll, ini, chm 등의 설치 관련 파일에 대해 설명한다.
- 폴더 및 설치 프로그램 실행 파일에 대해 설명한다.
- 설치 과정 및 결과가 기록되는 log 폴더에 대해 설명한다.

exe : 실행 가능한(executable) 파일의 확장자
dll : 장치의 드라이버 등 프로그램 설치 과정에서 필요한 경우 호출해서 사용하는 동적 링크 라이브러리 (dynamic link library) 파일의 확장자
ini : Windows 기반 컴퓨터의 기본 구성 값을 변경 해야 하는 경우 사용되는 설정 초기화 파일의 확장자
chm : HTML로 구성된 도움말 파일의 확장자
log : 프로그램이 실행되는 과정에서 발생하는 오류나 작업 결과 등이 기록된 파일로 향후 문제 발생 시 이를 진단하기 위한 자료로 사용됨

3. 제품 소프트웨어 패키징-SEC_04(소프트웨어 설치 매뉴얼 작성)

2) 서문

● 설치 환경 체크 항목

항목	내용
사용자 환경	CPU, Memory, OS(운영체제) 등
응용 프로그램	설치 전 다른 응용 프로그램 종료
업그레이드 버전	업그레이드 이전 버전에 대한 존재 유무 확인
백업 폴더 확인	데이터 저장 폴더를 확인하여 설치 시 폴더를 동기화시킴

3) 기본 사항

; 소프트웨어와 관련하여 기본적으로 설명되어야 할 항목들은 다음과 같다.

항목	설명
소프트웨어 개요	•소프트웨어의 주요 기능 및 UI 설명 •UI 및 화면 상의 버튼, 프레임 등을 그림으로 설명
설치 관련 파일	•소프트웨어 설치에 필요한 파일 설명 •exe, ini, log 등의 파일 설명
설치 아이콘(Installation)	설치 아이콘 설명
프로그램 삭제	설치된 소프트웨어의 삭제 방법 설명
관련 추가 정보	•소프트웨어 이외의 관련 설치 프로그램 정보 •소프트웨어 제작사 등의 추가 정보 기술

3. 제품 소프트웨어 패키징-SEC_04(소프트웨어 설치 매뉴얼 작성)

4) 설치 매뉴얼 작성 방법

; 설치 매뉴얼은 사용자가 설치 과정을 이해하기 쉽도록 설치 화면을 누락 없이 캡처하고 순서대로 상세히 설명한다.

● 설치 매뉴얼에는 설치 화면 및 UI, 설치 이상 메시지, 설치 완료 및 결과, FAQ, 설치 시 점검 사항, Network 환경 및 보안, 고객 지원 방법, 준수 정보 및 제한 보증 등에 대한 내용을 기술한다.

● 설치 화면 UI

- 설치 실행과 메인 화면 및 안내창에 대한 내용을 기술한다.
- 설치 실행 : exe 등의 설치(Install) 파일을 실행할 수 있도록 관련 실행 화면에 대한 이미지를 첨부하여 설명한다.
- 메인 화면 및 안내창 설치 시 나타나는 메인 화면과 각 과정에서의 안내창에 대한 이미지를 첨부하여 설명한다.

● 설치 이상 메시지 설명

- 설치 방법이나 설치 환경이 잘못된 경우 표시될 수 있는 메시지에 대해 설명한다.

예) 시스템 이상이나 xx 파일 오류 등으로 인해 해당 항목에 대한 설치를 진행할 수 없습니다.

1. 해당 원인 및 메시지에 대한 설명과 사용자의 이해를 돕기 위한 오류 코드표를 첨부한다.
2. 설치 이상 메시지 처리 과정에 대한 화면 이미지를 첨부한다.

3. 제품 소프트웨어 패키징-SEC_04(소프트웨어 설치 매뉴얼 작성)

4) 설치 매뉴얼 작성 방법

- 설치 완료 및 결과
  - 설치 완료 화면을 수록하여 설치가 정상적으로 마무리되었음을 사용자에게 최종적으로 알려준다.
- FAQ
  - 설치 과정에서 사용자가 직면할 수 있는 문제 상황에 대비할 수 있도록, 설치 시 발생할 수 있는 다양한 상황을 FAQ로 정리하여 수록한다.
- 설치 시 점검 사항
  - 설치 전 사용자의 설치 환경에 따라 점검해야 할 사항들이 무엇인지 설명한다.
  - 설치에 필요한 사용자 계정 및 설치 권한에 대해 확인할 수 있도록 설명한다.
  - 설치 과정에서 오류가 발생할 경우 점검할 수 있는 사항들에 대해 설명한다.

3. 제품 소프트웨어 패키징-SEC_04(소프트웨어 설치 매뉴얼 작성)

4) 설치 매뉴얼 작성 방법

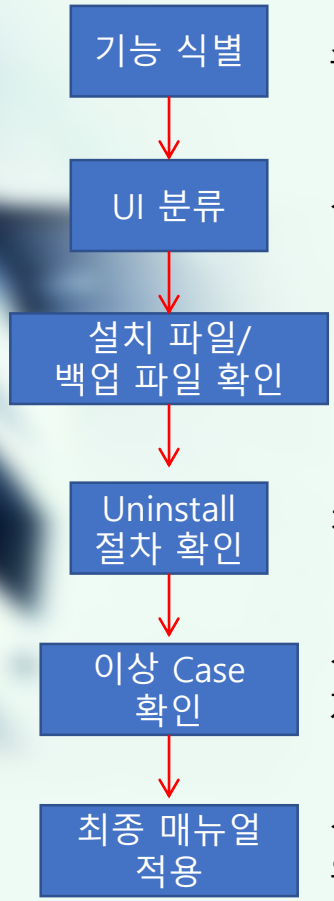
- Network 환경 및 보안
  - 네트워크 오류로 인해 설치 시 문제가 발생하지 않도록 사전에 필요한 네트워크 연결 상태를 점검하도록 안내한다.
  - 보안이나 방화벽으로 인해 설치 시 문제가 발생하지 않도록 관련된 내용을 안내한다.
- 고객 지원 방법(Customer Support)
  - 설치와 관련하여 기술적인 지원이나 소프트웨어에 대한 서비스를 원할 경우 국가, 웹 사이트, 전화번호, 이메일 등 문의할 수 있는 연락처를 안내한다.
- 준수 정보 & 제한 보증(Compliance Information & Limited Warranty)
  - Serial 보존, 불법 등록 사용 금지 등에 대한 준수 사항을 안내한다.
  - 저작권자 소유권 정보, SW 허가권 정보, 통신 규격, 개발 언어, 연동 프로그램, 문서 효력, 지적 소유권 정보 등과 관련된 내용을 안내한다.

Serial은 소프트웨어를 구별하기 위해 할당된 일련의 고유한 번호로 숫자 또는 숫자와 문자가 혼합되어 구성된다.

3. 제품 소프트웨어 패키징-SEC_04(소프트웨어 설치 매뉴얼 작성)

5) 설치 매뉴얼 작성 순서

; 소프트웨어 설치 매뉴얼은 다음과 같은 순서로 작성한다.



소프트웨어의 개발 목적과 주요 기능을 흐름 순으로 정리하여 기록한다.

설치 매뉴얼을 작성할 순서대로 UI를 분류한 후 기록한다.

폴더 위치, 설치 파일, 백업 파일 등의 개별적인 기능을 확인하여 기록한다.

직접 Uninstall을 수행하면서 그 순서를 단계별로 자세히 기록한다.

설치 과정에서 발생할 수 있는 다양한 Case를 만들어 확인하고 해당 Case에 대한 대처법 을 자세하게 기록한다.

설치가 완료된 화면과 메시지를 캡처하여 추가한다.
완성된 매뉴얼을 검토하고 고객 지원에 대한 내용을 기록한다.

3. 제품 소프트웨어 패키징-SEC_04(소프트웨어 설치 매뉴얼 작성) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(소프트웨어 설치 매뉴얼 작성)

1. 소프트웨어 설치 매뉴얼에 대한 설명으로 틀린 것은?

① 설치 과정에서 표시될 수 있는 예외 상황에 관련 내용을 별도로 구분하여 설명한다.

② 설치 시작부터 완료할 때까지의 전 과정을 빠짐없이 순서대로 설명한다.

③ 설치 매뉴얼은 개발자 기준으로 작성한다.

④ 설치 매뉴얼에는 목차, 개요, 기본 사항 등이 기본적으로 포함되어야 한다.

소프트웨어 설치 매뉴얼은 개발 초기에서부터 적용된 기준이나 사용자가 소프트웨어를 설치하는 과정에 필요한 내용을 기록한 설명서와 안내서이다.

설치 매뉴얼은 사용자 기준으로 작성을 한다.

설치 시작부터 완료할 때까지 전 과정을 빠짐없이 순서대로 설명한다.

설치 과정에서 표시될 수 있는 오류 메시지 및 예외 상황에 관한 내용을 별도로 분류하여 설명한다.

소프트웨어 설치 매뉴얼에는 목차 및 개요, 서문, 기본 사항 등이

3. 소프트웨어 설치 매뉴얼 작성에 대한 설명으로 잘못된 것은?

① 소프트웨어 설치 매뉴얼은 개발 초기에서부터 적용된 기준이나 사용자가 소프트웨어를 설치하는 과정에 필요한 내용을 기록한 설명서와 안내서이다.

② 소프트웨어 설치 매뉴얼에는 목차 및 개요, 서문, 기본 사항 등이 기본적으로 포함되어야 한다.

③ 일반적으로 서문에는 문서 이력, 설치 매뉴얼의 주석, 설치 도구의 구성, 설치 환경 체크 항목을 기술한다.

④ 설치 과정에서 표시될 수 있는 오류 메시지 및 예외 상황에 관한 내용을 별도로 분류하지 않고 관련 내용에 포함하여 설명한다.

설치 과정에서 표시될 수 있는 오류 메시지 및 예외 상황에 관한 내용은 별도로 분류하여 설명한다.

4. 서문에 작성할 내용으로 틀린 것은?

① 문서 이력　② 사용자 매뉴얼의 주석

③ 기록 보관 내용　④ 패치 내역

서문에는 문서 이력, 사용자 매뉴얼의 주석(주의 사항, 참고 사항), 기록 보관을 위해 필요한 내용을 기술한다.

패치는 이미 제작되어 릴리즈된 프로그램의 오류 수정이나 성능 향상을

3. 제품 소프트웨어 패키징-SEC_04(소프트웨어 설치 매뉴얼 작성) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(소프트웨어 설치 매뉴얼 작성)

5. 다음 중, 개발이 완성된 소프트웨어를 출시, 즉 배포하는 것을 말하는 용어는 무엇인가?

① 릴리즈 ② 제품 번호

③ Serial ④ 크랙

제품 번호 : 릴리즈 번호(Rev), 날짜 등 고유한 제품 번호를 기재하면 되는데 소프트웨어의 고유한 시리얼 넘버(Serial Number)를 기재한다.

Serial : 소프트웨어를 구별하기 위해 할당된 일련의 고유한 번호 숫자 또는 숫자와 문자가 혼합되어 구성됨

릴리즈(Release) : '풀어놓다'라는 의미로 개발이 완성된 소프트웨어를 출시, 즉 배포하는 것을 의미한다.

6. 다음 중, 설치 매뉴얼에 포함되지 않아도 되는 것은?

① 문서 이력

② 설치 화면 및 UI

③ 고객 지원 방법

④ FAQ

문서 이력은 서문에 기술한다.

3. 제품 소프트웨어 패키징-SEC_05(소프트웨어 사용자 매뉴얼 작성)

1) 소프트웨어 사용자 매뉴얼 작성의 개요

; 소프트웨어 사용자 매뉴얼은 사용자가 소프트웨어를 사용하는 과정에서 필요한 내용을 문서로 기록한 설명서와 안내서이다.

- 사용자 매뉴얼은 사용자가 소프트웨어 사용에 필요한 절차, 환경 등의 제반 사항이 모두 포함되도록 작성한다.
- 소프트웨어 배포 후 발생될 수 있는 오류에 대한 패치나 기능에 대한 업그레이드를 위해 매뉴얼의 버전을 관리한다.
- 개별적으로 동작이 가능한 컴포넌트 단위로 매뉴얼을 작성한다.
- 사용자 매뉴얼은 컴포넌트 명세서와 컴포넌트 구현 설계서를 토대로 작성한다.
- 사용자 매뉴얼에는 목차 및 개요, 서문, 기본 사항 등이 기본적으로 포함되어야 한다.
- 사용자 매뉴얼의 목차에는 매뉴얼 전체 내용을 순서대로 요약한 후 관련 내용의 시작 페이지를 함께 기술한다.
- 사용자 매뉴얼의 개요에는 소프트웨어의 주요 특징, 매뉴얼의 구성과 실행 방법, 사용법, 항목별 점검 기준, 항목별 설정 방법 등에 대한 내용을 기술한다.

컴포넌트(Component) : 컴포넌트는 독립적인 업무 또는 기능을 수행하는 단위이며, 실행 코드 기반으로 작성된 모듈이다.
컴포넌트 명세서 : 컴포넌트 명세서는 컴포넌트의 개요 및 내부 클래스의 동작, 외부와의 통신 명세 등을 정의한 문서이다.
컴포넌트 설계서 : 컴포넌트 설계서는 컴포넌트 구현에 필요한 컴포넌트 구조도 컴포넌트 목록, 컴포넌트 명세, 인터페이스 명세로 구성된 설계서이다.

3. 제품 소프트웨어 패키징-SEC_05(소프트웨어 사용자 매뉴얼 작성)

2) 서문

; 서문에는 문서 이력, 사용자 매뉴얼의 주석, 기록 보관을 위해 필요한 내용을 기술한다.

● 문서 이력

버전	작성자	작성일	검토자	일시	검수인
v0.1	신은혁	2014-05-20	황태자	2014-05-25	박인식
변경 내용	최초 작성				
v1.1	신은혁	2014-06-20	황태자	2014-06-25	박인식
변경 내용	제품 외형 변경				

● 설치 매뉴얼의 주석 : 주의 사항과 참고 사항을 기술한다.

- 주의 사항 : 소프트웨어를 사용할 때 사용자가 반드시 알고 있어야 하는 중요한 내용을 기술한다.
- 참고 사항 : 특별한 사용자의 환경이나 상황에 대한 내용을 기술한다.

● 기록 보관 내용

- 소프트웨어를 사용하면서 필요한 기술 지원이나 추가 정보를 얻기 위한 소프트웨어 등록 정보를 기술한다.
- 소프트웨어 등록 시 필요한 정보는 소프트웨어 명칭, 모델명, 문서 번호, 제품번호, 구입 날짜 등이다.

3. 제품 소프트웨어 패키징-SEC_05(소프트웨어 사용자 매뉴얼 작성)

3) 기본 사항

; 소프트웨어와 관련하여 기본적으로 설명되어야 할 항목들은 다음과 같다.

항목	설명
소프트웨어 개요	•소프트웨어의 주요 기능 및 UI 설명 •UI 및 화면 상의 버튼, 프레임 등을 그림으로 설명
소프트웨어 사용 환경	•소프트웨어 사용을 위한 최소 환경 설명 •CPU, 메모리 등의 PC 사양, 운영체제(OS) 버전 설명 •최초 구동에 대한 설명 •소프트웨어 사용 시 발생할 수 있는 프로그램 충돌이나 개인정보 보안 등에 관한 주의사항을 설명한다.
소프트웨어 관리	소프트웨어의 사용 종료 및 관리 등에 관한 내용 설명
모델, 버전별 특징	모델 및 버전별로 UI 및 기능의 차이점을 간략하게 요약한다.
기능, 인터페이스의 특징	제품의 기능과 인터페이스의 특징을 간략하게 요약한다.
소프트웨어 구동 환경	•개발에 사용한 언어 및 호환 가능한 운영체제(OS)에 대해 설명한다. •설치 후 구동하기까지의 과정을 운영체제(OS)별로 설명한다.

3. 제품 소프트웨어 패키징-SEC_05(소프트웨어 사용자 매뉴얼 작성)

4) 사용자 매뉴얼 작성 방법

; 사용자 매뉴얼은 사용자가 사용 방법을 이해하기 쉽도록 상황 별로 누락 없이 캡처하여 순서대로 상세히 설명한다.

- 사용자 매뉴얼에는 사용자 화면 및 UI, 주요 기능 분류, 응용 프로그램 및 설정, 장치 연동, Network 환경, Profile 안내, 고객 지원 방법, 준수 정보 및 제한 보증 등에 대한 내용을 기술한다.
- 사용자 화면 및 UI
  - 주의 사항과 참고 사항을 기술한다.
  - 주의사항 : 소프트웨어를 사용할 때 사용자가 반드시 알고 있어야 하는 중요한 내용을 설명한다.
  - 참고사항 : 특별한 사용자의 환경이나 상황에 대한 내용을 설명한다.
- 주요 기능 분류
  - 기능이 실행되는 화면을 순서대로 캡처하여 기능에 대한 사용법을 설명한다.
  - 기능이 구현되는 과정에서 참고할 사항이나 주의할 사항에 대한 메모를 추가한다.

3. 제품 소프트웨어 패키징-SEC_05(소프트웨어 사용자 매뉴얼 작성)

4) 사용자 매뉴얼 작성 방법

- 응용 프로그램(Program) 및 설정(Setting)
  - 소프트웨어 구동 시 함께 실행해도 되는 응용 프로그램, 또는 함께 실행되면 안 되는 응용 프로그램에 대해 설명한다.
  - 소프트웨어가 구동될 때 먼저 실행되어야 할 응용 프로그램이 있다면 설명한다.
  - 소프트웨어가 정상적으로 구동되기 위한 설정(Setting)이나 기본값에 대해 설명한다.
- 장치 연동
  - 소프트웨어가 특정 장치(Device)에 내장되는 경우 연동되는 장치(Device)에 대해 설명한다.
- Network 환경
  - Network에 접속되어 사용되는 소프트웨어인 경우 정상적인 연결을 위한 설정 값 등을 설명한다.
- Profile 안내
  - Profile은 소프트웨어의 구동 환경을 점검하는 파일로, 사용자가 Profile의 경로를 변경하거나 위치를 이동하지 않도록 안내한다.
  - Profile과 같이 소프트웨어 구동에 필수적인 파일에 대해 설명한다.

3. 제품 소프트웨어 패키징-SEC_05(소프트웨어 사용자 매뉴얼 작성)

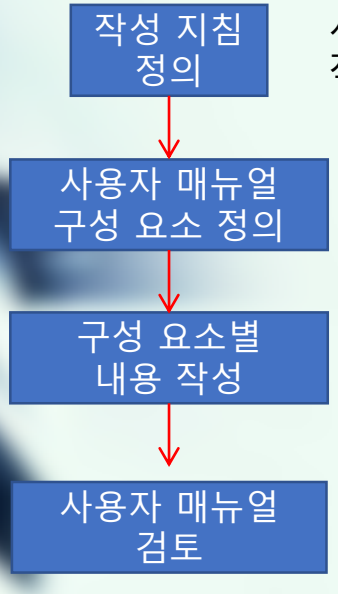
4) 사용자 매뉴얼 작성 방법

- 고객 지원 방법(Customer Support)
  - 사용과 관련하여 기술적인 지원이나 소프트웨어에 대한 서비스를 원할 경우 국가, 웹 사이트, 전화번호, 이메일 등 문의할 수 있는 연락처를 안내한다.
- 준수 정보 & 제한보증(Compliance Information & Limited Warranty)
  - Serial 보존, 불법 등록 사용 금지 등에 대한 준수 사항을 안내한다.
  - 저작권자 소유권 정보, SW 허가권 정보, 통신 규격, 개발 언어, 연동 프로그램, 문서 효력, 지적 소유권 정보 등과 관련된 정보를 안내한다.

3. 제품 소프트웨어 패키징-SEC_05(소프트웨어 사용자 매뉴얼 작성)

5) 사용자 매뉴얼 작성 순서

; 소프트웨어 사용자 매뉴얼은 다음과 같은 순서로 작성한다.



사용자 매뉴얼을 작성하기 위한 지침을 기록한다.
작성 지침은 사용자 환경에 필요한 정보를 제공할 수 있는 형태로 기록한다.

소프트웨어의 기능, 구성 객체 목록, 객체별 메소드, 메소드의 파라미터, 실제 사용 예,
사용자 환경 셋팅 방법 등을 기록한다.

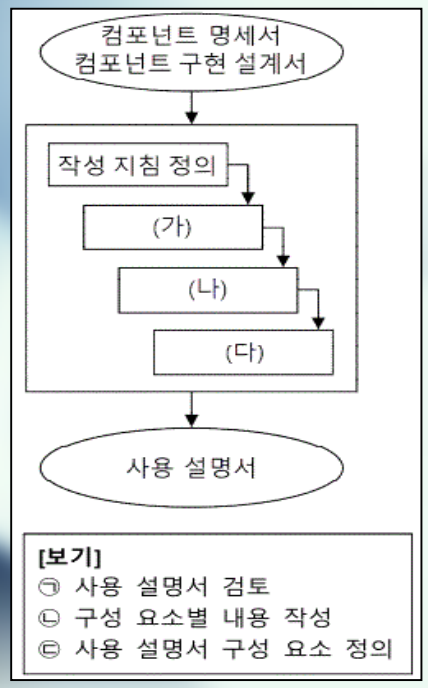
사용자 매뉴얼 구성 요소별로 내용을 기록한다.

작성된 구성 요소별 내용이 올바른지, 부족한 부분은 없는지 등을 검토하여 수정 및 보완한다.

3. 제품 소프트웨어 패키징-SEC_05(소프트웨어 사용자 매뉴얼 작성) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(소프트웨어 사용자 매뉴얼 작성)

1. 제품 소프트웨어의 사용자 매뉴얼 작성 절차로 (가)~(다)와 [보기]의 기호를 바르게 연결한 것은?



① (가)-㉠, (나)-㉡, (다)-㉢

② (가)-㉢, (나)-㉡, (다)-㉠

③ (가)-㉠, (나)-㉢, (다)-㉡

④ (가)-㉢, (나)-㉠, (다)-㉡

1. 작성 지침 정의 : 사용자 매뉴얼을 작성하기 위한 지침을 기록

2. 소프트웨어 사용자 매뉴얼에 대한 설명으로 잘못된 것은?

① 사용자 매뉴얼은 사용자가 설치와 사용에 필요한 절차, 환경 등의 제반 사항 모두가 포함되도록 작성한다.

② 사용자가 기술 지원을 받기 위해 소프트웨어를 등록할 때 소프트웨어 명, 소프트웨어 모델명, 제품 번호, 구입 날짜 등을 기재할 수 있도록 관련 내용을 사용자 매뉴얼에 포함한다.

③ 개별적으로 동작이 가능한 컴포넌트 단위로 매뉴얼이 작성되어야 한다.

④ 소프트웨어 구동 환경에 대한 내용은 해당 소프트웨어에 가장 최적화된 운영체제만을 대상으로 설명한다.

소프트웨어 사용자의 환경은 매우 다양하므로 매뉴얼에는 다양한 환경들에 대한 정보가 모두 포함되도록 작성한다.

소프트웨어 배포 후 발생될 수 있는 오류에 대한 패치, 기능에 대한 업그레이드를 위해 매뉴얼의 버전을 관리한다.

개별적으로 동작이 가능한 컴포넌트 단위로 매뉴얼을 작성한다.

사용자 매뉴얼은 컴포넌트 명세서와 컴포넌트 구현 설계서를 토대로 작성한다.

사용자 매뉴얼에는 목차 및 개요, 서문, 기본 사항 등이 기본적으로 포함되어야 한다.

3. 제품 소프트웨어 패키징-SEC_05(소프트웨어 사용자 매뉴얼 작성) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(소프트웨어 사용자 매뉴얼 작성)

4. 다음 중, 서문의 기록 보관 내용에 기재해야 하는 것이 아닌 것은?

① 소프트웨어 명칭

② 소프트웨어 가격

③ 제품 번호

④ 모델명

기록 보관 내용

소프트웨어를 사용하면서 필요한 기술 지원이나 추가 정보를 얻기 위한 소프트웨어 등록 정보를 기술한다.

소프트웨어 등록 시 필요한 정보는 소프트웨어 명칭, 모델명, 문서 번호, 제품 번호, 구입 날짜 등이다.

5. 다음 중, '컴포넌트의 개요 및 내부 클래스의 동작, 외부와의 통신 명세 등을 정의한 문서'를 무엇이라고 하는가?

① 패치

② 컴포넌트

③ 컴포넌트 명세서

④ 컴포넌트 설계서

컴포넌트 : 컴포넌트는 독립적인 업무 또는 기능을 수행하는

6. 다음 중, 사용자 매뉴얼에는 기술하는 내용이 아닌 것은?

① 장치 연동

② Profile 안내

③ Network 환경

④ 통신 규약

사용자 매뉴얼에는 사용자 화면 및 UI, 주요 기능 분류, 응용 프로그램 및 설정, 장치 연동, Network 환경, Profile 안내, 고객 지원 방법, 준수 정보 및 제한 보증 등에 대한 내용을 기술한다.

3. 제품 소프트웨어 패키징-SEC_06(소프트웨어 버전 등록)

1) 소프트웨어 패키징의 형상 관리

; 형상 관리(SCM : Software Configuration Management)는 소프트웨어의 개발 과정에서 소프트웨어의 변경 사항을 관리하기 위해 개발된 일련의 활동이다.

- 소프트웨어 변경의 원인을 알아내고 제어하며, 적절히 변경되고 있는지 확인하여 해당 담당자에게 통보한다.
- 형상 관리는 소프트웨어 개발의 전 단계에 적용되는 활동이며, 유지보수 단계에서도 수행된다.
- 형상 관리는 소프트웨어 개발의 전체 비용을 줄이고, 개발 과정의 여러 방해 요인이 최소화되도록 보증하는 것을 목적으로 한다.
- 관리 항목에는 소스 코드뿐만 아니라 프로젝트 계획, 분석서, 설계서, 프로그램, 테스트 케이스 등이 포함된다.
- 형상 관리를 통해 가시성과 추적성을 보장함으로써 소프트웨어의 생산성과 품질을 높일 수 있다.
- 대표적인 형상 관리 도구에는 Git, CVS, SVN 등이 있다.

형상 : 소프트웨어 개발 단계의 각 과정에서 만들어지는 프로그램, 프로그램을 설명하는 문서 데이터 등을 통칭하는 말이다.
Git : Git은 컴퓨터 파일의 변경사항을 추적하고 여러 명의 사용자들 간에 해당 파일들의 작업을 조율하기 위한 분산 버전 관리 시스템이다
CVS(Concurrent Versions System) : 특수한 저장소에서 동일한 소프트웨어 프로젝트의 다른 버전을 관리하도록 설계된 오픈 소스 소프트웨어 구성 관리 유틸리티이다.
SVN은 SubVersion의 줄임말로 중앙 집중 관리식 형상관리 소스 관리 툴이다.

3. 제품 소프트웨어 패키징-SEC_06(소프트웨어 버전 등록)

2) 형상 관리의 중요성

- 지속적인 소프트웨어의 변경 사항을 체계적으로 추적하고 통제할 수 있다.
- 제품 소프트웨어에 대한 무절제한 변경을 방지할 수 있다.
- 제품 소프트웨어에서 발견된 버그나 수정 사항을 추적할 수 있다.
- 소프트웨어는 형태가 없어 가시성이 결핍되므로 진행 정도를 확인하기 위한 기준으로 사용될 수 있다.
- 소프트웨어의 배포본을 효율적으로 관리할 수 있다.
- 소프트웨어를 여러 명의 개발자가 동시에 개발할 수 있다.

가시성(Visibility) : 일반적으로 가시성이란 대상을 확인할 수 있는 정도를 의미한다.

3. 제품 소프트웨어 패키징-SEC_06(소프트웨어 버전 등록)

3) 형상 관리 기능

; 형상 관리는 품질 보증을 위한 중요한 요소로서 다음과 같은 기능을 수행한다.

- **형상 식별** : 형상 관리 대상에 이름과 관리 번호를 부여하고, 계층(Tree) 구조로 구분하여 수정 및 추적이 용이하도록 하는 작업
- **버전 제어** : 소프트웨어 업그레이드나 유지 보수 과정에서 생성된 다른 버전의 형상 항목을 관리하고, 이를 위해 특정 절차와 도구(Tool)를 결합시키는 작업
- **형상 통제(변경 관리)** : 식별된 형상 항목에 대한 변경 요구를 검토하여 현재의 기준선(Base Line)이 잘 반영될 수 있도록 조정하는 작업
- **형상 감사** : 기준선의 무결성을 평가하기 위해 확인, 검증, 검열 과정을 통해 공식적으로 승인하는 작업
- **형상 기록(상태 보고)** : 형상의 식별, 통제, 감사 작업의 결과를 기록, 관리하고 보고서를 작성하는 작업

기준선(Base Line, 변경 통제 시점) : 기준선은 정식으로 검토되고 합의된 명세서나 제품으로, 소프트웨어 개발 시 소프트웨어 변경을 적절히 제어할 수 있도록 도와준다.

무결성은 결점이 없다는 것으로, 정해진 기준에 어긋나지 않고 조건을 충실히 만족하는 정도라고 이해할 수 있다.

3. 제품 소프트웨어 패키징-SEC_06(소프트웨어 버전 등록)

4) 소프트웨어의 버전 등록 관련 주요 기능

; 소프트웨어 개발 과정에서 코드와 라이브러리, 관련 문서 등의 버전을 관리하기 위해 자료를 등록하고 갱신하는 과정에서 사용되는 주요 용어와 의미는 다음과 같다.

항목	설명
저장소(Repository)	최신 버전의 파일들과 변경 내역에 대한 정보들이 저장되어 있는 곳이다.
가져오기(Import)	버전 관리가 되고 있지 않은 아무것도 없는 저장소(Repository)에 처음으로 파일을 복사한다.
체크아웃(Check-Out)	•프로그램을 수정하기 위해 저장소(Repository)에서 파일을 받아온다. •소스 파일과 함께 버전 관리를 위한 파일들도 받아온다.
체크인(Check-In)	체크아웃 한 파일의 수정을 완료한 후 저장소(Repository)의 파일을 새로운 버전으로 갱신한다.
커밋(Commit)	체크인을 수행할 때 이전에 갱신된 내용이 있는 경우에는 충돌(Conflict)을 알리고 diff 도구를 이용해 수정한 후 갱신을 완료한다.
동기화(Update)	저장소에 있는 최신 버전으로 자신의 작업 공간을 동기화한다.

diff 도구 : diff 도구는 비교 대상이 되는 파일들의 내용(소스 코드)을 비교하며 서로 다른 부분을 찾아 표시해 주는 도구이다.

3. 제품 소프트웨어 패키징-SEC_06(소프트웨어 버전 등록)

5) 소프트웨어 버전 등록 과정

; 소프트웨어의 버전 등록은 다음과 같은 순서로 진행한다.

개발자가 저장소에 신규로 파일을 추가한다.

수정 작업을 진행할 개발자가 저장소에 추가된 파일을 자신의 작업 공간으로 인출한다.

인출한 파일을 수정한 후 설명을 붙여 저장소에 예치한다.

커밋(Commit) 후 새로운 개발자가 자신의 작업 공간을 동기화(Update)한다.
이때 기존 개발자가 추가했던 파일이 전달된다.

새로운 개발자가 추가된 파일의 수정 기록(Change Log)을 확인하면서 이전 개발자가 처음 추가한 파일과 이후 변경된 파일의 차이를 확인한다.

버전 관리 프로그램에 따라 방법은 다를 수 있지만, diff <commit> <commit2>와 같이 지정하면, 지정한 두 커밋(Commit) 사이의 수정 내역을 확인할 수 있다. 이와 같이 이전 개발자들의 수정 내역을 확인하고 싶을 때 diff 명령을 사용한다

3. 제품 소프트웨어 패키징-SEC_06(소프트웨어 버전 등록) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(소프트웨어 버전 등록)

1. 소프트웨어의 개발 과정에서 소프트웨어의 변경 사항을 관리 하기 위해 개발된 일련의 활동을 뜻하는 것은?

① 복호화 ② 형상 관리

③ 저작권 ④ 크랙

형상관리(SCM : Software Configuration Management)는 소프트웨어의 개발 과정에서 소프트웨어의 변경 사항을 관리하기 위해 개발된 일련의 활동을 의미한다.

복호화 또는 디코딩(decoding)은 부화화(encoding)된 데이터를 부호(code)화 되기 전 형태로 바꾸어, 사람이 읽을 수 있는 형태로 되돌려 놓는 것이다.

저작권이란 컴퓨터 프로그램 저작물 등에 대하여 창작자가 가지는 배타적 독점적 권리로 타인의 침해를 받지 않을 고유한 권리이다.

크랙이란 '깨다'라는 의미 그대로 불법적인 방법으로 소프트웨어에 적용된 저작권 보호 기술을 해제하여 무단으로 사용할 수 있도록 하는 기술이나 도구를 의미한다.

2. 소프트웨어 형상 관리에서 관리 항목에 포함되지 않는 것은?

① 프로젝트 요구 분석서

3. 제품 소프트웨어의 형상 관리 역할로 틀린 것은?

① 형상 관리를 통해 이전 리버전이나 버전에 대한 정보에 접근 가능 하여 배포본 관리에 유용

② 불필요한 사용자의 소스 수정 제한

③ 프로젝트 개발 비용을 효율적으로 관리

④ 동일한 프로젝트에 대해 여러 개발자 동시 개발 가능

지속적인 소프트웨어의 변경 사항을 체계적으로 추적하고 통제할 수 있다.

제품 소프트웨어에 대한 무절제한 변경을 방지할 수 있다.

제품 소프트웨어에서 발견된 버그나 수정 사항을 추적할 수 있다.

소프트웨어는 형태가 없어 가시성이 결핍되므로 진행 정도를 확인하기 위한 기준으로 사용될 수 있다.

소프트웨어의 배포본을 효율적으로 관리할 수 있다.

소프트웨어를 여러 명의 개발자가 동시에 개발할 수가 있다.

4. 형상관리 도구의 주요기능으로 거리가 먼 것은?

① 정규화(Normalization)

② 체크인(Check-in)

③ 체크아웃(Check-out)

3. 제품 소프트웨어 패키징-SEC_06(소프트웨어 버전 등록) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(소프트웨어 버전 등록)

5. 소프트웨어 형상 관리에 대한 설명으로 거리가 먼 것은?

① 소프트웨어에 가해지는 변경을 제어하고 관리한다.

② 프로젝트 계획, 분석서, 설계서, 프로그램, 테스트 케이스 모두 관리 대상이다.

③ 대표적인 형상 관리 도구로 Ant, Maven, Gradle 등이 있다.

④ 유지 보수 단계뿐만 아니라 개발 단계에도 적용할 수 있다.

Ant, Maven, Gradle은 빌드 자동화 도구에 해당한다.

소프트웨어 변경의 원인을 알아내고 제어하며, 적절히 변경되고 있는지 확인하여 해당 담당자에게 통보한다.

형상 관리는 소프트웨어의 개발의 전 단계에 적용되는 활동이며, 유지보수 단계에서도 수행된다.

형상 관리는 소프트웨어 개발의 전체 비용을 줄이고, 개발 과정의 여러 방해 요인을 최소화되도록 보증하는 것을 목적으로 한다.

관리 항목에는 프로젝트 계획, 분석서, 설계서, 프로그램, 테스트 케이스 모두 관리 대상이다.

형상 관리를 통해 가시성과 추적성을 보장함으로써 소프트웨어의 생산성과 품질을 높일 수 있다.

7. 소프트웨어 형상 관리(Configuration Management)에 관한 설명으로 틀린 것은?

① 소프트웨어에서 일어나는 수정이나 변경을 알아내고 제어하는 것을 의미한다.

② 소프트웨어 개발의 전체 비용을 줄이고, 개발 과정의 여러 방해 요인이 최소화되도록 보증하는 것을 목적으로 한다.

③ 형상 관리를 위하여 구성된 팀을 "Chief Programmer Team"이라고 한다.

④ 형상 관리의 기능 중 하나는 버전 제어 기술이다.

Chief Programmer Team은 효율성을 증대시키기 위해 경험과 능력이 풍부한 책임 프로그래머를 중심으로 구성한 개발팀의 구성 방식 중 하나로 형상 관리와는 관계가 없다.

8. 다음 중, 형상 관리는 품질 보증을 위한 중요한 요소로서 '식별된 형상 항목에 대한 변경 요구를 검토하여 현재의 기준선(Base Line)이 잘 반영될 수 있도록 조정하는 작업'을 무엇이라고 하는가?

① 형상 식별　　② 버전 제어

③ 형상 통제(변경 관리)　　④ 형상 감사

형상 식별 : 형상 관리 대상에 이름과 관리 번호를 부여하고, 계층 구조로

3. 제품 소프트웨어 패키징-SEC_07(소프트웨어 버전 관리 도구)

1) 공유 폴더 방식

; 공유 폴더 방식은 버전 관리 자료가 로컬 컴퓨터의 공유 폴더에 저장되어 관리되는 방식으로, 다음과 같은 특징이 있다.

- 개발자들은 개발이 완료된 파일을 약속된 공유 폴더에 매일 복사한다.
- 담당자는 공유 폴더의 파일을 자기 PC로 복사한 후 컴파일 하여 이상 유무를 확인한다.
- 이상 유무 확인 과정에서 파일의 오류가 확인되면, 해당 파일을 등록한 개발자에게 수정을 의뢰한다.
- 파일에 이상이 없다면 다음날 각 개발자들이 동작 여부를 다시 확인한다.
- 파일을 잘못 복사하거나 다른 위치로 복사하는 것에 대비하기 위해 파일의 변경 사항을 데이터베이스에 기록하여 관리한다.
- 종류에는 SCCS, RCS, QVCS 등이 있다.

RCS(Revision Control System) : 여러 개발자가 프로젝트를 수행할 때 시간에 따른 파일 변화 과정을 관리하는 소프트웨어 버전 관리 도구로 소스 파일을 동시에 수정하는 것을 방지하고 다른 방향으로 진행된 개발 결과를 합치거나 변경 내용을 추적할 수 있다.
QVCS : Quma Software에서 발표한 버전 제어 시스템 제품군이다

3. 제품 소프트웨어 패키징-SEC_07(소프트웨어 버전 관리 도구)

2) 클라이언트/서버 방식

; 클라이언트/서버 방식은 버전 관리 자료가 중앙 시스템(서버)에 저장되어 관리되는 방식으로, 다음과 같은 특징이 있다.

- 서버의 자료를 개발자 별로 자신의 PC(클라이언트)로 복사하여 작업한 후 변경된 내용을 서버에 반영한다.
- 모든 버전 관리는 서버에서 수행된다.
- 하나의 파일을 서로 다른 개발자가 작업할 경우 경고 메시지를 출력한다.
- 서버에 문제가 생기면, 서버가 복구되기 전까지 다른 개발자와의 협업 및 버전 관리 작업은 중단된다.
- 종류에는 CVS, SVN(Subversion), Clear Case, Perforce 등이 있다.

CVS(Concurrent Versions System) : 특수한 저장소에서 동일한 소프트웨어 프로젝트의 다른 버전을 관리하도록 설계된 오픈 소스 소프트웨어 구성 관리 유틸리티이다.
SVN은 SubVersion의 줄임말로 중앙 집중 관리식 형상관리 소스 관리 툴이다.
CVSNT : 협업 프로그래밍을 하는 서버
Clear Case : 모든 종류의 파일과 디렉터리에 대한 버전 관리 기능을 제공한다.
Perforce : 소스 버전 관리 툴의 일종이다.

3. 제품 소프트웨어 패키징-SEC_07(소프트웨어 버전 관리 도구)

3) 분산 저장소 방식

; 분산 저장소 방식은 버전 관리 자료가 하나의 원격 저장소와 분산된 개발자 PC의 로컬 저장소에 함께 저장되어 관리되는 방식으로, 다음과 같은 특징이 있다.

- 개발자 별로 원격 저장소의 자료를 자신의 로컬 저장소로 복사하여 작업한 후 변경된 내용을 로컬 저장소에서 우선 반영(버전 관리)한 다음 이를 원격 저장소에 반영한다.
- 로컬 저장소에서 버전 관리가 가능하므로 원격 저장소에 문제가 생겨도 로컬 저장소의 자료를 이용하여 작업할 수 있다.
- 종류에는 Git, GNU arch 등이 있다.

Git : Git은 컴퓨터 파일의 변경사항을 추적하고 여러 명의 사용자들 간에 해당 파일들의 작업을 조율하기 위한 분산 버전 관리 시스템이다.
GNU arch : 분산형 리비전 관리 소프트웨어의 일종으로 소스 트리에서 일어나는 변경들을 추적하며 통합을 비롯해 여러 사람에 의해 또는 서로 다른 여러 시간대에 일어나는 다양한 변화들을 처리하는 도구이다.

3. 제품 소프트웨어 패키징-SEC_07(소프트웨어 버전 관리 도구)

4) Subversion(서브버전, SVN)

; Subversion은 CVS를 개선한 것으로, 아파치 소프트웨어 재단에서 2000년에 발표하였다.

- 클라이언트/서버 구조로, 서버(저장소, Repository)에는 최신 버전의 파일들과 변경 내역이 관리된다.
- 서버의 자료를 클라이언트로 복사해와 작업한 후 변경 내용을 서버에 반영(Commit)한다.
- 모든 개발 작업은 trunk 디렉터리에서 수행되며, 추가 작업은 branches 디렉터리 안에 별도의 디렉터리를 만들어 작업을 완료한 후 trunk 디렉터리와 병합(merge)한다.
- 커밋(Commit)할 때마다 리비전(Revision)이 1씩 증가한다.
- 클라이언트는 대부분의 운영체제에서 사용되지만, 서버는 주로 유닉스를 사용한다.
- 소스가 오픈 되어 있어 무료로 사용할 수 있다.
- CVS의 단점이었던 파일이나 디렉터리의 이름 변경, 이동 등이 가능하다.

trunk : trunk는 '몸통', '줄기'라는 의미로 개발 과정에서 가장 중심이 되는 디렉터리이다. trunk 디렉터리 안에 소스 파일과 추가 작업을 위한 서브 디렉터리인 branches 디렉터리가 있다.
branches : '가지', '부문'이라는 의미로 메인 개발 과정과는 별도로 새로운 기능의 테스트와 같이 추가적인 작업을 수행하기 위한 디렉터리이다. branches 디렉터리 하위에 작업 별로 디렉터리를 만들어 그 안에서 개발을 진행한다. 이후 별도의 디렉터리 에서 진행된 개발 결과를 trunk와 병합할 수 있다.
리비전 : 커밋의 버전으로, 처음 저장소를 만들면 리비전은 0이 된다. 이후 커밋이 수행될 때마다 리비전이 1씩 증가한다.

3. 제품 소프트웨어 패키징-SEC_07(소프트웨어 버전 관리 도구)

4) Subversion(서브버전, SVN)

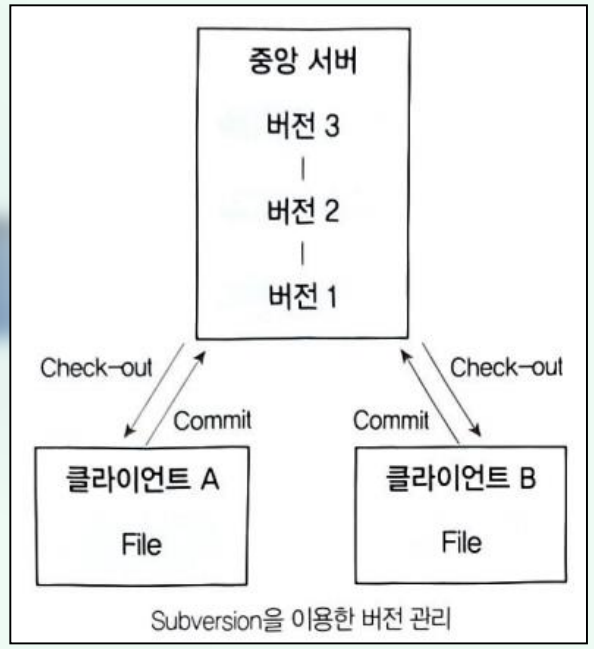
● 다음은 Subversion의 주요 명령어이다.

명령어	의미
add	•새로운 파일이나 디렉터리를 버전 관리 대상으로 등록한다. •add로 등록되지 않은 대상은 commit이 적용되지 않는다.
commit	버전 관리 대상으로 등록된 클라이언트의 소스 파일을 서버의 소스 파일에 적용한다.
update	•서버의 최신 commit 이력을 클라이언트의 소스 파일에 적용한다. •commit 전에는 매번 update를 수행하여 클라이언트에 적용되지 않은 서버의 변동 내역을 클라이언트에 적용한다..
checkout	버전 관리 정보와 소스 파일을 서버에서 클라이언트로 받아온다.
lock/unlock	서버의 소스 파일이나 디렉터리를 잠그거나 해제한다.
import	아무것도 없는 서버의 저장소에 맨 처음 소스 파일을 저장하는 명령으로 한 번 사용하면 다시 사용하지 않는다.
export	버전 관리에 대한 정보를 제외한 순수한 소스 파일만을 서버에서 받아온다.
info	지정한 파일에 대한 위치나 마지막 수정 일자 등에 대한 정보를 표시한다.
diff	지정된 파일이나 경로에 대해 이전 리비전과의 차이를 표시한다.
merge	다른 디렉터리에서 작업된 버전 관리 내역을 기본 개발 작업과 병합한다.

3. 제품 소프트웨어 패키징-SEC_07(소프트웨어 버전 관리 도구)

4) Subversion(서브버전, SVN)

- Subversion을 이용한 버전 관리



Subversion을 이용한 버전 관리

Subversion을 이용해 버전 관리 작업을 시작할 때는 먼저 'import' 명령으로 모든 소스 파일을 서버에 등록한다. 이후 버전 관리는 'checkout' -> 작업 -> add -> update -> commit 과정으로 진행한다. 나머지 명령은 작업 과정이나 자료 송수신 과정 에서 필요에 의해 수행된다.

3. 제품 소프트웨어 패키징-SEC_07(소프트웨어 버전 관리 도구)

5) Git(깃)

; Git은 리누스 토발즈(Linus Torvalds)가 2005년 리눅스 커널 개발에 사용할 관리 도구로 개발한 이후 주니오 하마노(Junio Hamano)에 의해 유지 보수되고 있다.

- Git은 분산 버전 관리 시스템으로 2개의 저장소, 즉 지역(로컬) 저장소와 원격 저장소가 존재한다.
- 지역 저장소는 개발자들이 실제 개발을 진행하는 장소로, 버전 관리가 수행된다.
- 원격 저장소는 여러 사람들이 협업을 위해 버전을 공동 관리하는 곳으로, 자신의 버전 관리 내역을 반영하거나 다른 개발자의 변경 내용을 가져올 때 사용한다.
- 버전 관리가 지역 저장소에서 진행되므로 버전 관리가 신속하게 처리되고, 원격 저장소나 네트워크에 문제가 있어도 작업이 가능하다.
- 브랜치를 이용하면 기본 버전 관리 틀에 영향을 주지 않으면서 다양한 형태의 기능 테스팅이 가능하다.
- 파일의 변화를 스냅샷(Snapshot)으로 저장하는데, 스냅샷은 이전 스냅샷의 포인터를 가지므로 버전의 흐름을 파악할 수 있다.

브랜치(Branch) : Git에서는 저장소가 처음 만들어지면 마스터(Mater) 브랜치가 생성되고 브랜치에서 기본적인 버전 관리가 진행된다. 새로운 기능을 추가하는 작업은 새로운 브랜치를 만들어 작업을 수행하며, 작업이 정상적으로 마무리되면 작업 내역 을 마스터 브랜치에 병합한다. 이렇게 마스터 브랜치와 별도로 생성하는 브랜치를 토픽(Topic) 브랜치 또는 피처(Feature) 브랜 라고 한다. 각각의 브랜치는 다른 브랜치에 영향을 주지 않으므로 독립적인 여러 작업을 동시에 진행할 수 있다.
스냅샷(Snapshot) : 스냅샷은 영문자와 숫자가 혼합된 40자리 문자열로 표시된다.

3. 제품 소프트웨어 패키징-SEC_07(소프트웨어 버전 관리 도구)

5) Git(깃)

● 다음은 Git의 주요 명령어이다.

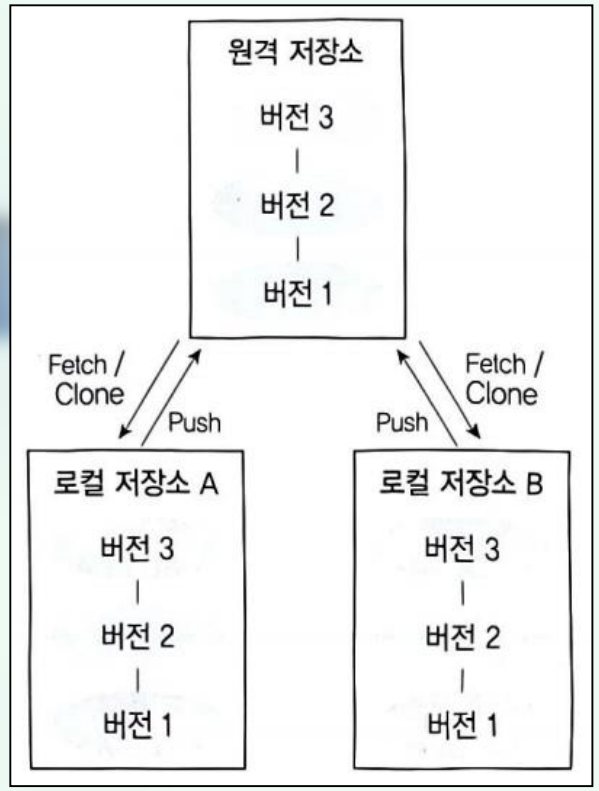
명령어	의미
add	•작업 내역을 지역 저장소에 저장하기 위해 스테이징 영역(Staging Area)에 추가한다. •'--all' 옵션으로 작업 디렉터리의 모든 파일을 스테이징 영역에 추가할 수 있다.
commit	작업 내역을 지역 저장소에 저장한다.
branch	•새로운 브랜치를 생성한다. •최초로 commit을 하면 마스터(master) 브랜치가 생성된다. •commit 할 때마다 해당 브랜치는 가장 최근의 commit한 내용을 가리키게 된다. •'d' 옵션으로 브랜치를 삭제할 수 있다.
checkout	•지정한 브랜치로 이동한다. •현재 작업 중인 브랜치는 HEAD 포인터가 가리키는데, checkout 명령을 통해 HEAD 포인터를 지정한 브랜치로 이동시킨다.
merge	지정한 브랜치의 변경 내역을 현재 HEAD 포인터가 가리키는 브랜치에 반영함으로써 두 브랜치를 병합한다.
init	지역 저장소를 생성한다.
remote add	원격 저장소에 연결한다.
push	로컬 저장소의 변경 내역을 원격 저장소에 반영한다.
fetch	원격 저장소의 변경 이력만을 지역 저장소로 가져와 반영한다.

스테이징(Staging) 영역 : 작업 내역을 바로 commit 지역 저장소에 저장하지 않고 스테이징 영역에 저장했다가 commit을 하는 이유는 스테이징 영역에서 작업 내용을 한 번 더 확인하여 선별적으로 지역 저장소에 반영하기 위함이다. 이렇게 하면 스테이징 영역을 사용하지 않을 때보다 시간은 더 소요되지만 좀 더 안정된 버전 관리 작업이 가능하다.

3. 제품 소프트웨어 패키징-SEC_07(소프트웨어 버전 관리 도구)

5) Git(깃)

● Git을 이용한 버전 관리



Git을 이용해 버전 관리 작업을 시작할 때는 먼저 'init' 명령으로 지역 저장소를 만들고, 'remote add' 명령으로 원격 저장소 에 연결한 후 'add --all -> commit -> push'를 한다. 이후 버전 관리는 'fetch -> 작업 -> add -> commit -> push' 과정으로 진행한다. 나머지 명령은 작업 과정이나 자료 송수신 과정에서 필요에 의해 수행된다.

3. 제품 소프트웨어 패키징-SEC_07(소프트웨어 버전 관리 도구) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(소프트웨어 버전 관리 도구)

1. 다음 설명의 소프트웨어 버전 관리 도구방식은?

버전 관리 자료가 원격 저장소와 로컬 저장소에 함께 저장되어 관리된다.
로컬 저장소에서 버전 관리가 가능하므로 원격 저장소에 문제가 생겨도 로컬 저장소의 자료를 이용하여 작업할 수 있다.
대표적인 버전 관리 도구로 Git이 있다.

① 단일 저장소 방식
② 분산 저장소 방식
③ 공유 폴더 방식
④ 클라이언트/서버 방식

1. 단일 저장소 방식(Monorepo) : 하나의 저장소를 사용하는 방식 모든 프로젝트의 버전 기록이 프로젝트와 관련된 모든 사람들에게 의미가 있다면, 하나의 저장소로 유지하는 게 좋다.
2. 분산 저장소 방식 : 버전 관리 자료가 원격 저장소와 로컬 저장소에 함께 저장되어 관리된다. 로컬 저장소에서 버전 관리가 가능하므로 원격 저장소에 문제가 생겨도 로컬 저장소의 자료를 이용하여 작업할

3. 다음은 버전 관리 도구인 Subversion에 대한 설명이다. 잘못된 것은?

① 클라이언트/서버 구조로, 서버에는 최신 버전과 버전의 변화를 저장한다.
② 클라이언트에서는 서버의 자료를 복사해와 작업한 후 변경된 내용을 서버에 반영(Commit)한다.
③ 모든 개발 작업은 trunk 디렉터리에서 수행되며, 부가적인 추가작업 은 branches 디렉터리 안에 별도의 디렉터리를 만들어 작업을 완료한 후 trunk 디렉터리의 작업과 병합한다.
④ 커밋(Commit)할 때마다 커밋의 버전이라고 할 수 있는 스냅샷(Snapshot)이 일정하게 증가한다.

Subversion은 CVS를 개선한 것으로, 아파치 소프트웨어 재단에서 개발하였다.
클라이언트/서버 구조로, 서버(저장소, Repository)에는 최신 버전의 파일들과 변경 내역이 관리된다.
서버의 자료를 클라이언트로 복사해와 작업한 후 변경 내용을 서버에 반영한다.
모든 개발 작업은 trunk 디렉터리에서 수행되며, 부가적인 추가작업 은

3. 제품 소프트웨어 패키징-SEC_07(소프트웨어 버전 관리 도구) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(소프트웨어 버전 관리 도구)

5. 다음은 버전 관리 도구인 Subversion에서 사용하는 명령어들에 대한 설명이다. 잘못된 것은?

① add : commit을 수행할 버전 관리 대상을 등록한다.

② update : 최신 commit 이력을 소스 파일에 적용한다.

③ export : 아무것도 없는 서버의 저장소에 맨 처음 소스 파일을 저장한다.

④ checkout : 서버에서 클라이언트로 버전 관리를 위한 내용과 소스 파일을 받아온다.

add : commit을 수행할 버전 관리 대상을 등록한다. add로 등록되지 않은 대상은 commit이 적용되지 않는다.

commit : 버전 관리 대상으로 등록된 클라이언트의 소스 파일을 서버에 소스 파일에 적용한다.

update : 최신 commit 이력을 소스 파일에 적용한다. commit 전에 매번 update를 수행하여 클라이언트에 적용되지 않은 서버의 변동 내역을 클라이언트에 적용한다.

import : 아무것도 없는 서버의 저장소에 맨 처음 소스 파일을 저장하고 한 번 사용하면 다시 사용하지 않는다.

3. 제품 소프트웨어 패키징-SEC_08(빌드 자동화 도구)

1) 빌드 자동화 도구의 개념

; 빌드란 소스 코드 파일들을 컴파일 한 후 여러 개의 모듈을 묶어 실행 파일로 만드는 과정이며, 이러한 빌드를 포함하여 테스트 및 배포를 자동화하는 도구를 빌드 자동화 도구라고 한다.

● 애자일 환경에서는 하나의 작업이 마무리될 때마다 모듈 단위로 나눠서 개발된 코드들이 지속적으로 통합되는데, 이러한 지속적인 통합(Continuous Integration)개발 환경에서 빌드 자동화 도구는 유용하게 활용된다.

● 빌드 자동화 도구에는 Ant, Make, Maven, Gradle, Jenkins 등이 있으며, 이중 Jenkins와 Gradle이 가장 대표적이다.

Ant : 자바 프로그래밍 언어에서 사용하는 자동화된 소프트웨어 빌드 도구
make : 유닉스 계열 운영 체제에서 사용되는 프로그램 빌드 도구이다.
Maven : Apache Ant와 Maven의 차이점은 Apache Ant는 소프트웨어 빌드 프로세스를 자동화하는 소프트웨어 도구이고 Maven은 소프트웨어 프로젝트 관리 도구라는 것이다. Maven은 소프트웨어 빌드 프로세스를 자동화하는 도구 그 이상이다. 전반적으로 Maven은 Ant보다 더 유연하다.
Gradle : 그루비(Groovy): 자바에 파이썬, 루비, 스몰토크 등의 특징을 더한 동적 객체 지향 프로그래밍 언어이며, Gradle은 그루비를 이용한 빌드 자동화 시스템이다.
Jenkins : JAVA 기반의 오픈 소스 형태로, 가장 많이 사용되는 빌드 자동화 도구이다.

3. 제품 소프트웨어 패키징-SEC_08(빌드 자동화 도구)

2) Jenkins

; Jenkins는 JAVA 기반의 오픈 소스 형태로, 가장 많이 사용되는 빌드 자동화 도구이다.

- 서블릿 컨테이너에서 실행되는 서버 기반 도구이다.
- SVN, Git 등 대부분의 형상 관리 도구와 연동이 가능하다.
- 친숙한 Web GUI 제공으로 사용이 쉽다.
- 여러 대의 컴퓨터를 이용한 분산 빌드나 테스트가 가능하다.

서블릿 컨테이너 : 서블릿 컨테이너는 클라이언트의 요청을 처리해 주기 위해 서버 측에서 실행되는 작은 프로그램(Server Side Applet)인 서블릿을 실행하고 서블릿의 생명주기를 관리하는 역할을 한다.
서블릿 : 클라이언트의 요청을 처리하고, 그 결과를 반환하는 Servlet 클래스의 구현 규칙을 지킨 자바 웹 프로그래밍 기술

3. 제품 소프트웨어 패키징-SEC_08(빌드 자동화 도구)

3) Gradle

; Gradle은 Groovy를 기반으로 한 오픈 소스 형태의 자동화 도구로, 안드로이드 앱 개발 환경에서 사용된다.

- 안드로이드 뿐만 아니라 플러그 인을 설정하면, JAVA, C/C++, Python 등의 언어도 빌드가 가능하다.
- Groovy를 사용해서 만든 DSL(Domain Specific Language)을 스크립트 언어로 사용한다.
- Gradle은 실행할 처리 명령들을 모아 태스크(Task)로 만든 후 태스크 단위로 실행한다.
- 이전에 사용했던 태스크를 재사용하거나 다른 시스템의 태스크를 공유할 수 있는 빌드 캐시 기능을 지원하므로 빌드의 속도를 향상시킬 수 있다.

DSL(Domain Specific Language) : DSL이란 웹 페이지 영역에 특화되어 사용되는 HTML과 같이 특정한 도메인, 즉 영역이나 용도 에 맞게 기능을 구성한 언어를 말한다.
스크립트 언어(Script Language) : 스크립트 언어는 HTML 문서 안에 직접 프로그래밍 언어를 삽입하여 사용하는 것으로 기계어 로 컴파일 되지 않고 별도의 번역기가 소스를 분석하여 동작하게 하는 언어이다.

3. 제품 소프트웨어 패키징-SEC_08(빌드 자동화 도구) 기출 및 출제 예상 문제

기출 및 출제 예상 문제(소프트웨어 버전 관리 도구)

1. 빌드 자동화 도구에 대한 설명으로 틀린 것은?

① Gradle은 실행할 처리 명령들을 모아 태스크로 만든 후 태스크 단위로 실행한다.

② 빌드 자동화 도구는 지속적인 통합 개발 환경에서 유용하게 활용된다.

③ 빌드 자동화 도구에는 Ant, Gradle, Jenkins 등이 있다.

④ Jenkins는 Groovy를 기반으로 한 오픈 소스로 안드로이드 앱 개발 환경에서 사용된다.

Jenkins는 JAVA를 기반으로 한 오픈 소스로 가장 많이 사용하는 빌드 자동화 도구이다.

서블릿 컨테이너에서 실행되는 서버 기반 도구이다.

SVN, Git 등 대부분의 형상 관리 도구와 연동이 가능하다.

친숙한 Web GUI를 제공한다.

여러 대의 컴퓨터를 이용한 분산 빌드나 테스트가 가능하다.

2. 대표적인 빌드 자동화 도구인 Jenkins와 Gradle에 대한 설명으로 잘못된 것은?

① 빌드, 테스트, 배포 과정을 자동화 하는 도구이다.

3. 다음 중, 빌드 자동화 도구에 해당하지 않는 것은?

① Ant ② Gradle

③ Jenkins ④ DSL

Ant : 자바 프로그래밍 언어에서 사용하는 자동화된 소프트웨어 빌드 도구

make : 유닉스 계열 운영 체제에서 사용되는 프로그램 빌드 도구이다.

Maven : 소프트웨어 프로젝트 관리 도구이다. Maven은 소프트웨어 빌드 프로세스를 자동화 하는 도구 그 이상이다. 전반적으로 Maven은 Ant보다 더 유연하다.

감사합니다.