제129회 정보관리기술사 해설집

2023.02.04

국가기술자격 기술사 시험문제

기술사 제 129 회 제 2 교시 (시험시간: 100 분)

분야	정보통신	자격종목	정보관리기술사	수검번호		성명	

※ 다음 문제 중 4 문제를 선택하여 설명하시오. (각 10 점)

1. 가상화(Virtualization)에 대하여 다음을 설명하시오.

   가. 일반적인 운영 체제의 프로그램 동작방식

   나. 응용 프로그램 가상화 동작방식

   다. 원격 데스크톱 프로토콜의 종류

2. 접근 제어(Access Control)의 통제정책과 경량 디렉토리 엑세스 프로토콜(LDAP: Lightweight Directory Access Protocol)의 인증흐름(Flow)에 대하여 설명하시오

3. 딥뷰(DeepView)의 개념과 기술요소를 설명하시오

4. 최근 대규모 공공 차세대 시스템이 오픈 이후에 많은 문제점이 발생되어 사회적 불편을 초래하게 되었다. 이에 대하여 다음을 설명 하시오.

가. 발생된 문제점의 원인

나. 재발 방지를 위한 대책 및 법제도 보완 방안

다. 시스템 오픈 가능 여부 판단을 위한 지표 관리

5. A 기업은 다수의 기존 정보시스템을 운영 및 유지보수를 하고 있으며 신규 시스템에 대한 개발을 기획 중에 있다. 개발 방법론으로 구조적 방법론을 주로 활용하여 왔지만 Agile 방법론의 도입을 검토하고 있다. 다음의 사항에 대하여 설명하시오.

가. 구조적 방법론과 Agile 방법론 비교

나. Agile 방법론의 스크럼(Scrum)과 칸반(Kanban)설명

다. Agile 방법론의 효율적인 수행 방안 제시

6. 객체지향 기법 중에는 리팩토링(Refactoring)과 디자인패턴(Design Pattern)이 있다. 두 기법을 각각 정의하고 공통점과 차이점에 대하여 설명하시오.

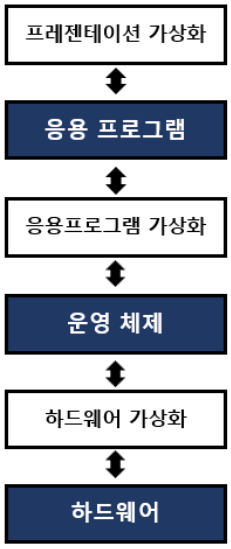
01	응용프로그램 가상화		
문제	가상화(Virtualization)에 대하여 다음을 설명하시오. 가. 일반적인 운영 체제의 프로그램 동작방식 나. 응용 프로그램 가상화 동작방식 다. 원격 데스크톱 프로토콜의 종류		
도메인	디지털서비스	난이도	중(상/중/하)
키워드	응용 프로그램 가상화, 프레젠테이션 가상화 RDP(Remote Desktop Protocol), Telnet Protocol, rlogin, PCoIP(PC over IP), ICA(Independent Computing Architecture), RFB(Remote Frame Buffer)		
출제배경	재택 근무, 보안강화 측면 활용이 증가하고 있는 응용프로그램 가상화에 대한 출제		
참고문헌	가상화 기술 분석 및 동향 연구(제34회 한국정보처리학회, 2010.11) 운영체제와 정보기술의 원리(반효경 저) 가상화의 기본 개념(정보문화사) 원격데스크톱(RDP) 터널링 공격 분석 및 대응방법(한국사회보장정보원, 2021.01)		
해설자	소원반 소민호 기술사(제 119회 정보관리기술사 / mhsope@naver.com)		

I. 시스템 자원의 추상화, 가상화(Virtualization)의 개요

가. 가상화(Virtualization)의 개념

- 물리적으로 다른 시스템을 논리적으로 통합하거나 하나의 시스템을 논리적으로 분할해 자원을 사용하게 하는 기술

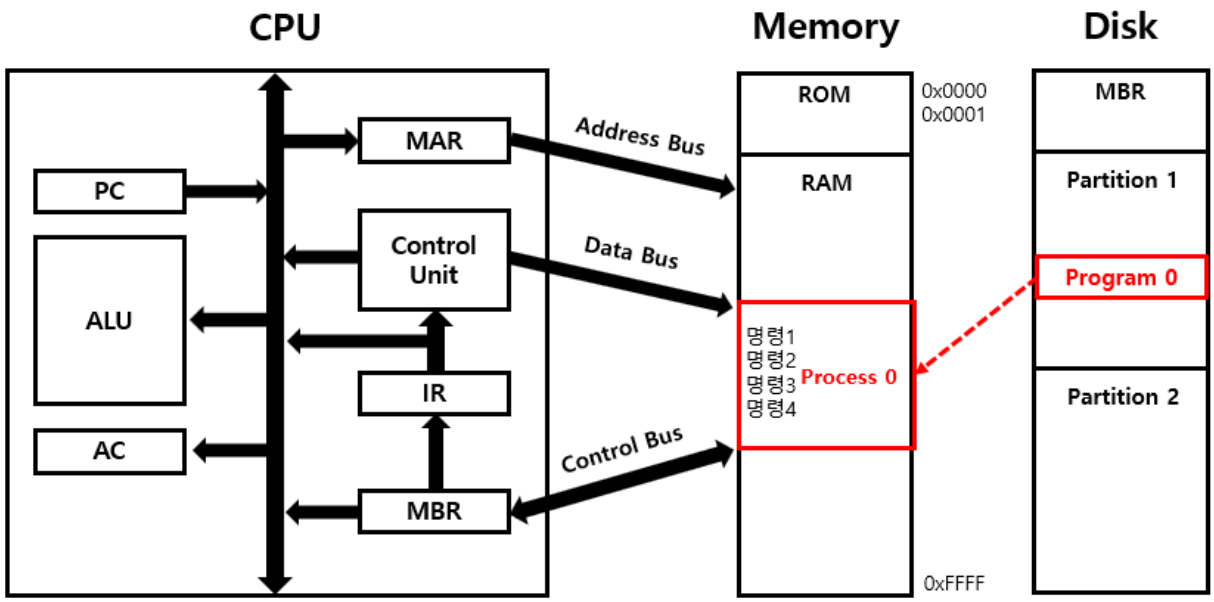
나. 가상화의 종류

개념도	가상화 종류	
	프레젠테이션 가상화	- 애플리케이션을 중앙 서버에 설치하고 가상의 인터페이스만 네트워크를 통해 보내는 기술 - 모든 작업을 서버에서 처리하고 그 화면(이미지)만 PC에 전송하는 방식
	응용 프로그램 가상화	- 응용 프로그램 구성 계층을 OS에서 분리시키는 기술 - 추상 계층(Abstract Layer)을 구성하고 이 위에 응용 프로그램이 올라가는 형태로 동작 - 추상의 계층에서 프로그램에 대한 고유의 환경을 구성하여 실제 운영 체제엔 영향을 주기 않음.
	하드웨어 가상화	- 소프트웨어를 사용하여 물리적 컴퓨터를 실행하는 가상 컴퓨터

- **응용 프로그램 가상화**는 복잡한 관리 감소, 유연한 확장성, 보안성 강화 측면 강점이 있음

II. 일반적인 운영체제의 프로그램 동작방식 설명

가. 일반적인 운영체제의 프로그램 동작 구성도



구성요소	설명
Register	- CPU 내부에 위치한 Access 속도가 가장 빠른 기억장치
PC (Program Counter)	- 다음에 실행할 명령의 메모리 주소를 가리키는 레지스터
ALU (Arthmetic Logic Unit)	- 각종 산술 연산들과 논리 연산들을 수행하는 회로들로 이루어진 하드웨어 모듈
AC (Accumulator)	- 일시적으로 저장하는 레지스터
MAR (Memory Address Register)	- PC에 저장된 명령어 주소가 시스템 주소 버스로 출력되기 전에 일시적으로 저장되는 주소 레지스터
CU (Control Unit)	- 명령어를 해독하여 데이터 경로가 데이터를 적절히 처리하도록 제어 신호를 발생시키는 장치
IR (Instruction Register)	- 가장 최근에 인출된 명령어 코드가 저장되어 있는 레지스터
MBR (Memory Buffer Register)	- 기억장치에 쓰여질 데이터 혹은 기억장치로부터 읽혀진 데이터를 일시적으로 저장하는 버퍼 레지스터
System Bus	- Address Bus : 주소 전송, CPU -> RAM (단방향) - Data Bus : 데이터 전송, CPU -> RAM (단방향) - Control Bus : 제어 신호 전송, CPU <-> RAM (양방향)

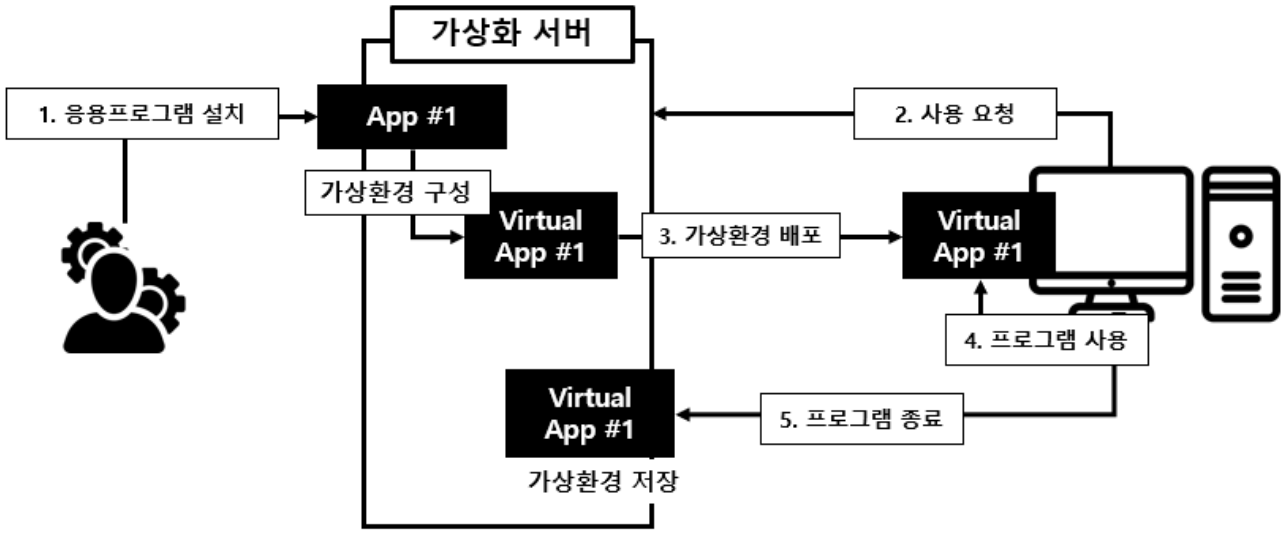
나. 일반적인 운영체제의 프로그램 동작방식 상세 설명

순서	동작	설명
1	프로그램 실행	- Disk에서 설치된 프로그램의 실행파일을 읽어 들임
2	프로세스 생성	- 새로운 프로세스 생성
3	메모리 적재	- 운영체제에 의해 프로그램을 구성하는 명령어와 데이터가 메모리에 적재 - 프로세스 주소 공간은 코드(code), 데이터(data), 스택(stack)등으로 구성 - 당장 수행에 필요한 부분은 메모리에 적재되고, 나머지는 디스크의 스왑영역에 저장
4	실행 대기	- 프로세스가 실행 순서를 기다림
5	CPU 할당	- 프로세스가 운영체제 스케줄링 방식에 의해 CPU를 할당 받고 명령 수행 준비
6	명령어 수행	- 프로세스의 주소 값이 낮은 곳부터 메모리를 읽어 명령을 수행 - 명령어 수행은 크게 fetch(인출)와 execution(실행)으로 구성

- 일반적으로 물리적인 H/W에 운영체제와 응용프로그램이 설치되어 동작
- 모든 응용 프로그램은 메모리 할당과 같은 서비스 수행을 위해 운영체제에 의존적

III. 응용 프로그램 가상화의 동작방식 설명

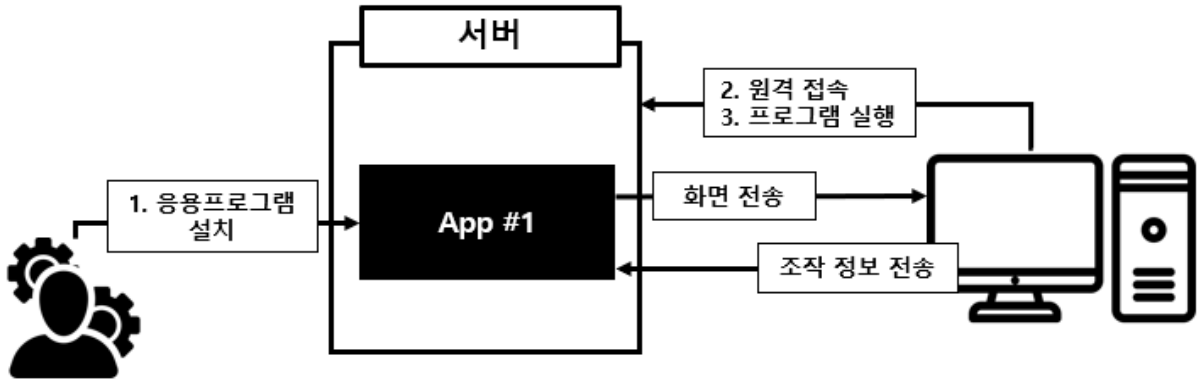
가. 응용 프로그램 가상화의 동작방식



순서	동작	설명
1	응용프로그램 설치	- 응용프로그램 가상화 서버에 응용 프로그램을 설치 및 가상환경 구성
2	사용 요청	- 사용자는 응용프로그램을 사용하기 위하여 사용요청함
3	가상환경 배포	- 사용자별 가상 환경(폴더, 파일, 레지스트리, 서비스 등) 구성 및 배포
4	프로그램 사용	- 사용자는 접속 프로그램을 활용 인증, 인가 후 응용프로그램을 사용
5	프로그램 종료	- 사용이 완료된 응용 프로그램은 고유 환경(폴더, 파일, 레지스트리, 서비스 등) 구성을 패키지 형태로 별도 저장 후 추후 재사용시 활용

- 응용프로그램 가상화는 플랫폼 위에, 추상 계층(Abstract Layer)가 있고, 이 위에 응용 프로그램이 올라가는 형태로 동작
- 추상의 계층에서 프로그램에 대한 고유의 환경을 구성하여 실제 운영 체제엔 영향을 주기 않음.

나. 프리젠테이션 가상화의 동작방식



순서	동작	설명
1	프로그램 설치	- 원격 서버에 응용 프로그램 설치
2	원격 접속	- 사용자는 원격 접속 프로그램(RDP)을 통해 원격 서버에 접속
3	프로그램 실행	- 원격 서버에 설치된 프로그램을 실행 및 작업 수행 - 모든 작업은 원격서버에서 처리되고 그 **화면(이미지)만 PC에 전송**

- 사용자가 애플리케이션을 사용하면서 키보드를 입력하거나 마우스를 클릭한 정보가 네트워크를 통해서 서버로 보내지게 되며, 화면은 사용자 디바이스에 업데이트된 정보를 전달하게 되어 실제로는 어떠한 데이터도 사용자 디바이스에서 저장되지 않는 형태

- 대표적으로 프레젠테이션 가상화 기술이 원격 데스크톱 프로토콜(Remote Desktop Protocol) 서비스 임.

IV. 원격 데스크톱 프로토콜의 종류

프로토콜	설명
RDP(Remote Desktop Protocol)	- 마이크로소프트에서 개발된 윈도우의 원격 데스크톱 전용 프로토콜 - TCP 3389 포트 사용, 다른 프로토콜보다 네트워크 부하가 적고 응답이 빠름
Telnet Protocol	- 사용자가 인터넷을 통해 원격 시스템에 접속해 제어를 가능하게 하는 프로토콜 - TCP 23 포트 사용, 네트워크 가상 단말(NVT)을 사용
rlogin	- Unix BSD계열 시스템 간의 원격접속을 위해서 설계된 프로토콜 - TCP 513 포트 사용
PCoIP(PC over IP)	- 원격 데스크톱과 어플리케이션을 엔드 포인트에게 전달하기 위한 원격 디스플레이 프로토콜 - VM웨어의 가상 데스크톱 시스템용 제품 VMware View에서 채용됨
ICA(Independent Computing Architecture)	- 윈도우 기반 터미널에서 윈도우 터미널 서버를 이용하기 위한 프로토콜 - 영국의 시트릭스 시스템즈 사가 윈도우 NT용으로 개발 - 네트워크 사용량이 적도록 설계되어 클라이언트 측에 부담을 주지 않고 이용 가능
RFB(Remote Frame Buffer)	- 그래픽 사용자 인터페이스에 대한 원격 액세스를 위한 개방형 단순 프로토콜 - TCP 5900 포트 사용, 모든 시스템 및 응용 프로그램에 적용

- 최근 RDP에 대한 다양한 취약점(CVE-2022-21893, CVE-2019-1181, CVE-2020-0609, CVE-2019-1182, CVE-2019-0708)이 발표됨에 따라 RDP에 대한 취약점 대응방안 필요

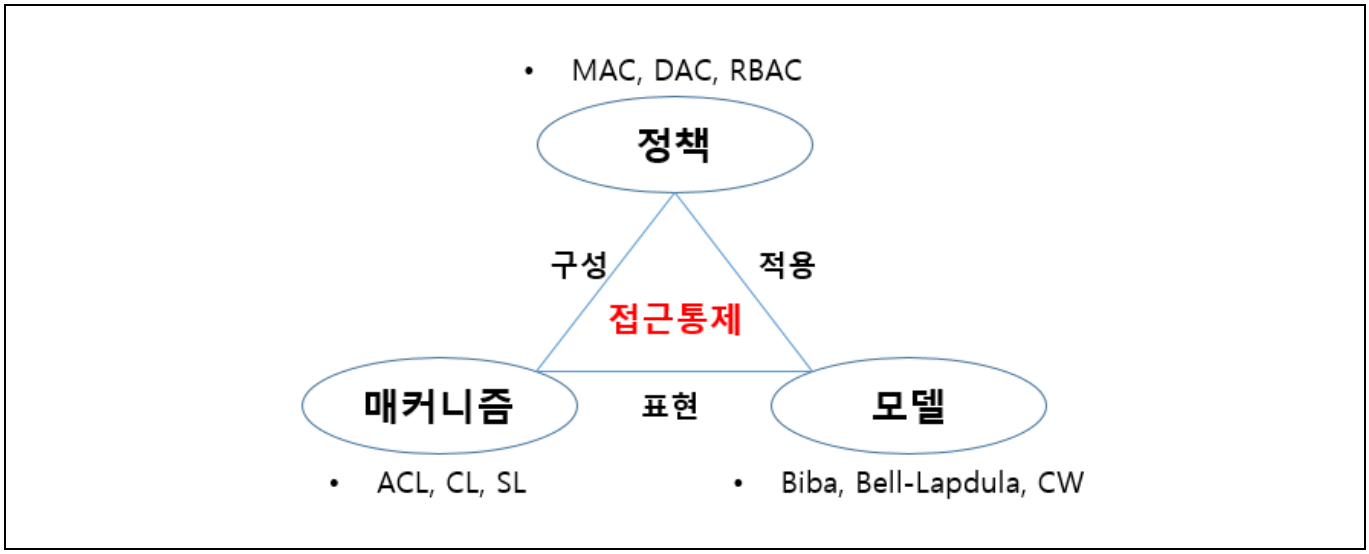
V. 원격 데스크톱 프로토콜 보안 취약점 대응방안

대응방안	설명
원격 데스크톱 서비스 비활성화	- 원격 연결에 서비스가 필요하지 않은 시스템에서 서비스 비활성화 - 인 바운드 RDP 연결을 명시적으로 거부하는 호스트 기반 방화벽 규칙 활성화 - 원격 데스크톱 서비스를 통한 로그온 거부 보안 설정을 활성화
강력한 비밀번호 사용	- RDP 액세스에 항상 강력한 사용자 이름과 비밀번호를 사용 - 관리자 계정에 이중 인증을 활성화하고 액세스를 항상 VPN 뒤에 숨긴
역할 기반 액세스 제한	- RDP 콘솔에 대한 관리자 액세스 사용자 수 제한 및 권한 제한
RDP NLA 활성화	- RDP의 NLA(Network Level Authentication)를 통한 추가 인증 활성화
RDP 포트 변경	- RDP 포트 변경을 통한 포트 스캐너가 감지 예방
RDP 서버 추적 및 로깅/모니터링 활성화	- 자사 시스템의 RDP 활성화 유무 파악 - RDP 서버 로그에서 로깅(Logging) 및 모니터링을 활성화
RDP 게이트웨이 활용	- 승인된 사용자만 RDP 사용, 액세스 장치 통제 가능

"끝"

02	접근제어(Access Control), LDAP(Lightweight Directory Access Protocol)		
문제	접근 제어(Access Control)의 통제정책과 경량 디렉토리 엑세스 프로토콜(LDAP: Lightweight Directory Access Protocol)의 인증흐름(Flow)에 대하여 설명하시오.		
도메인	정보보안	난이도	중(상/중/하)
키워드	MAC, DAC, RBAC, Log4J, X.500, 필터링		
출제배경	2021년 보안 취약점 Log4J와 관련된 LDAP 관련 기술 이해 확인		
참고문헌	소프트웨어 개발보안 가이드 (한국인터넷진흥원)		
해설자	강남평일야간반 전일 기술사(제 114회 정보관리기술사 / nikki6@hanmail.net)		

I. 인가된 주체만이 객체 접근, 접근 통제 개요



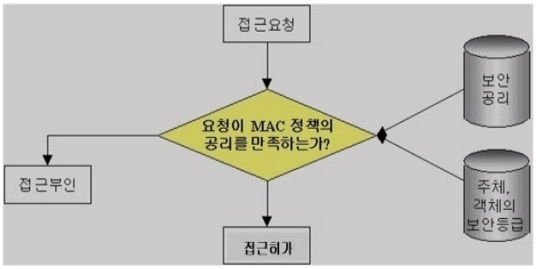
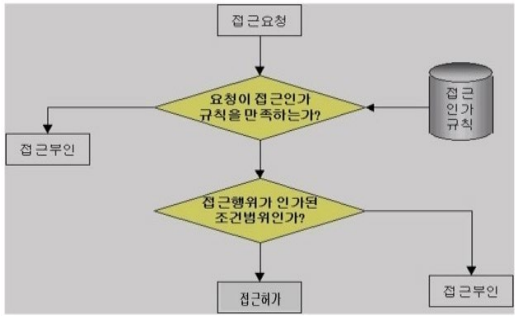
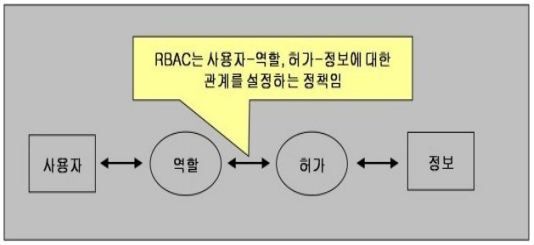
- 사용자(주체)의 신원을 식별/인증하여 대상정보(객체)의 접근, 사용수준을 인가(Authorization)하는 기법으로 접근통제 정책, 매커니즘, 모델로 구성

II. 접근 통제 정책

가. 접근통제 정책의 원칙

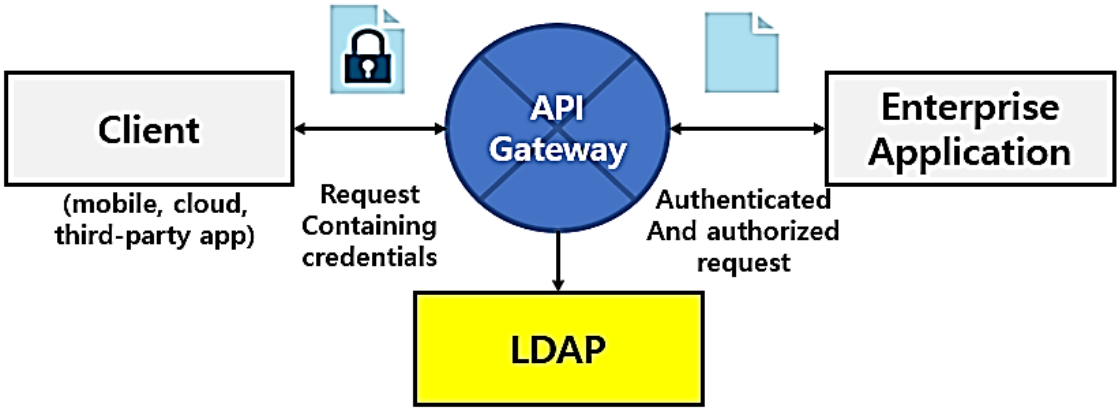
원칙	내용
최고 권한 부여	- 사용자들이 업무를 수행하기 위하여 꼭 필요한 권한만을 가지도록 접근 권한을 부여
최대 권한의 정책	- 데이터 공유의 장점을 증대 시키기 위하여 적용하는 최대 가용성 원리에 기반
직무 분리의 원칙	- 보안/감사, 개발/생산, 암호키 관리/변경 등 직무에 따라 접근 권한 분리

나. 접근통제 정책의 유형

유형	개념도	설명	주체
MAC (Mandatory Access Control)		- 강제적 접근통제, 규칙 기반 접근 통제, 관리자 - 사전 정의된 규칙(Rule)을 기반으로 주제(Subject)에게 허용된 접근 권한과 객체에 부여된 허용 등급을 비교하여 접근 통제	시스템
DAC (Discretionary Access Control)		- 임의적 접근통제, ACL(Access Control List), 사용자 소유권과 권한 - 주체의 신분에 근거하여 객체에 대한 접근 통제, 주체가 자신이 소유한 객체에 대해 다른 주체의 접근 권한 임의 설정 가능	사용자
RBAC (Role-Based Access Control)		- 역할 기반(임무 기반) 접근통제 - 사용자의 역할에 기반을 두고 접근을 통제하는 정책, 주체가 역할이라는 추상화 단계를 거쳐 객체의 접근 권한을 부여	사용자 역할

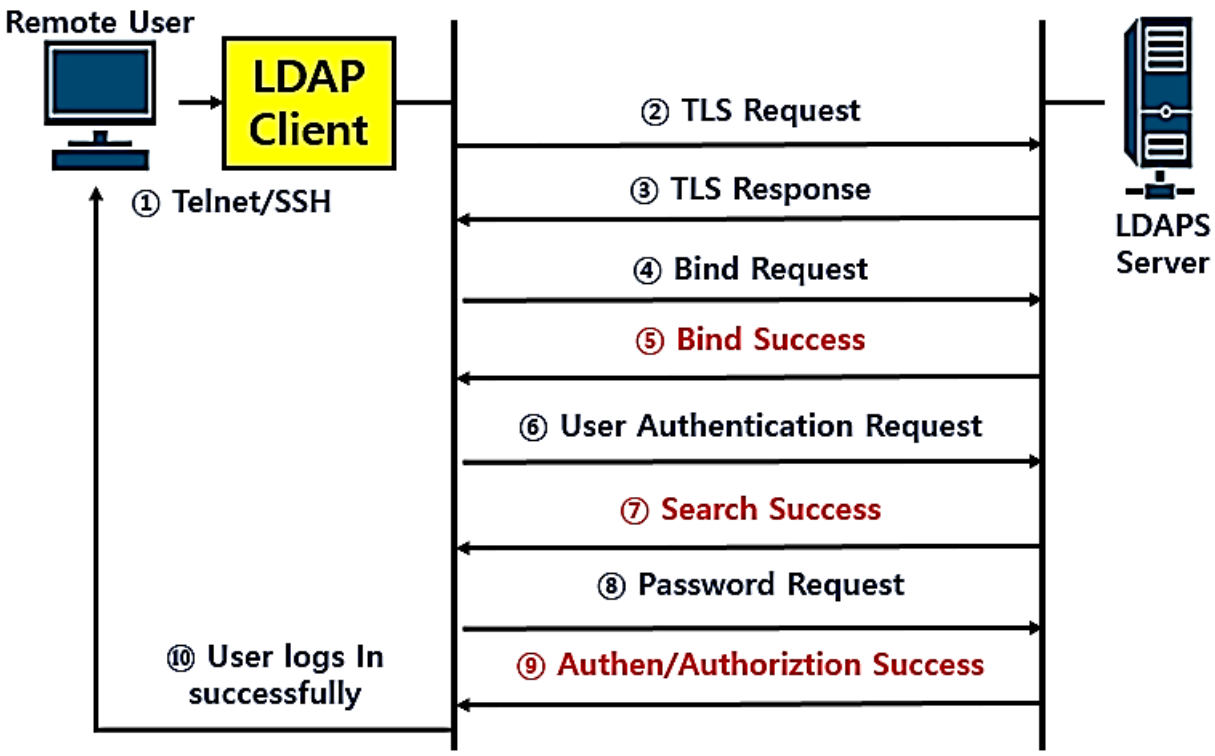
III. 경량 디렉토리 엑세스 프로토콜의 인증 흐름(Flow)

가. 경량 디렉토리 엑세스 프로토콜(LDAP: Lightweight Directory Access Protocol) 개념



- 네트워크 상에서 조직이나 개인정보 혹은 파일이나 디바이스 정보 등을 찾아보는 것을 가능하게 만든 소프트웨어 프로토콜
- 네트워크 상의 디렉토리 서비스 표준인 X.500의 DAP(Directory Access Protocol)를 기반으로 한 경량화(Lightweight) 된 DAP 버전

나. 디렉토리 엑세스 프로토콜(LDAP: Lightweight Directory Access Protocol)의 인증흐름(Flow)



Seq	세부내용
1	- 원격 사용자는 SSH, TELNET 또는 기타 로그인 유틸리티를 통해 디바이스에 로그인
2	- LDAPS 클라이언트는 TLS 프로토콜 요청을 사용하여 LDAPS 인증 서버와 TCP 연결을 구축
3	- 클라이언트가 TLS 응답을 수신하면 클라이언트와 서버가 ID를 인증
4	- LDAPS 클라이언트는 bind 요청(binddn 및 bindpw)을 사용하여 LDAPS 서버에서 사전 구성된 프록시 계정을 사용하여 스스로 인증
5	- bind 작업이 성공하면 LDAPS 서버는 LDAPS 클라이언트에 있음을 인정
6,7,8	- 그런 다음 LDAPS 클라이언트는 로그인하려고 하는 사용자의 로그인 자격 증명과 함께 인증 요청을 LDAPS 서버로 전송
9	- 인증이 성공하면 LDAPS 클라이언트가 사용자에게 로그인 성공에 대해 통보. 사용자의 권한 부여 데이터는 나중에 인증을 적용하는 데 사용되는 파일에 저장
10	- 클라이언트는 LDAPS 서버로 연결 종료

- 2021년 발생했던 역대급 보안 취약점 Log4J는 JNDI(Java Naming and Directory Interface)와 LDAP를 이용

IV. LDAP 삽입 취약점 대응방안

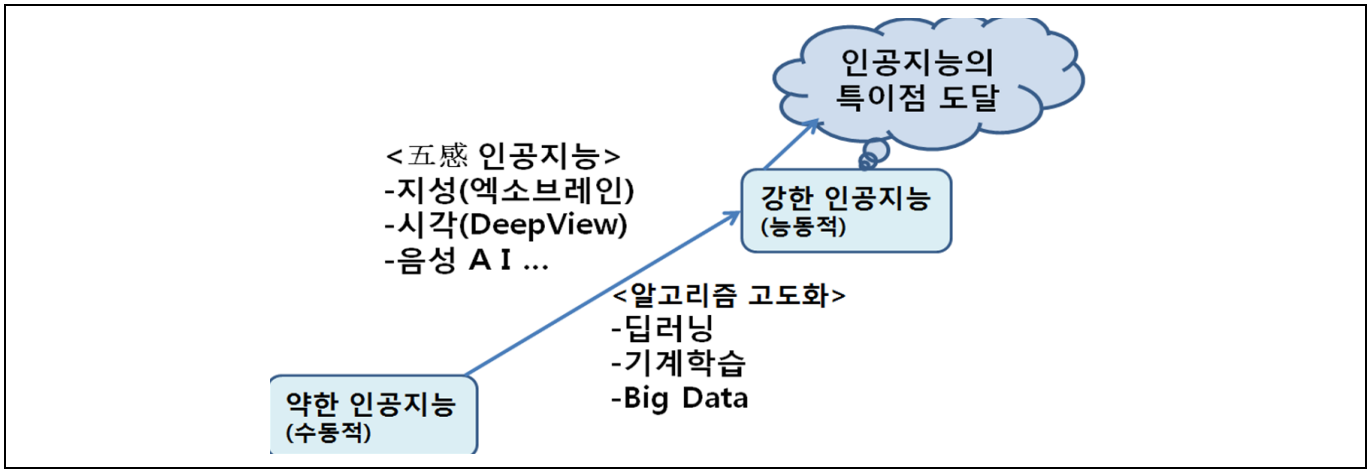
개발환경	활용 가능한 프레임워크 또는 라이브러리
Java, PHP, ASP	- LDAP Syntax Filters: http://social.technet.microsoft.com/wiki/contents/articles/5392.active-directory-ldap-syntax-filters.aspx - LDAP 검색필터로 액티브 디렉토리 조회 시의 검색기준을 정의하고 효율적인 검색을 수행 - 위 MS 웹 문서의 내용과 참조목록으로 필터 작성에 필요한 문법을 참고해 특수문자 필터링

- LDAP 인증이 포함되는 기능 설계 시, 외부 입력 값이 LDAP 조회를 위한 검색 필터 생성에 삽입되어 사용되는 경우, 필터 규칙으로 인식 가능한 특수문자(=, +, <, >, #, ;, ₩ 등)들을 제거하고 사용할 수 있도록 시큐어 코딩 규칙을 정의

"끝"

03	인공지능 딥뷰		
문제	딥뷰(DeepView)의 개념과 기술요소를 설명하시오		
도메인	인공지능	난이도	중(상/중/하)
키워드	시물인지, 시각데이터, 시멘틱 추론, 이미지 분석, 예측기술, 실시간 대응		
출제배경	딥뷰 활용 증가에 따른 기술 개념과 요소 확인		
참고문헌	ITPE 기술사회 자료집		
해설자	정상 기술사(제 12X회 정보관리기술사 / jeongsang_pe@naver.com)		

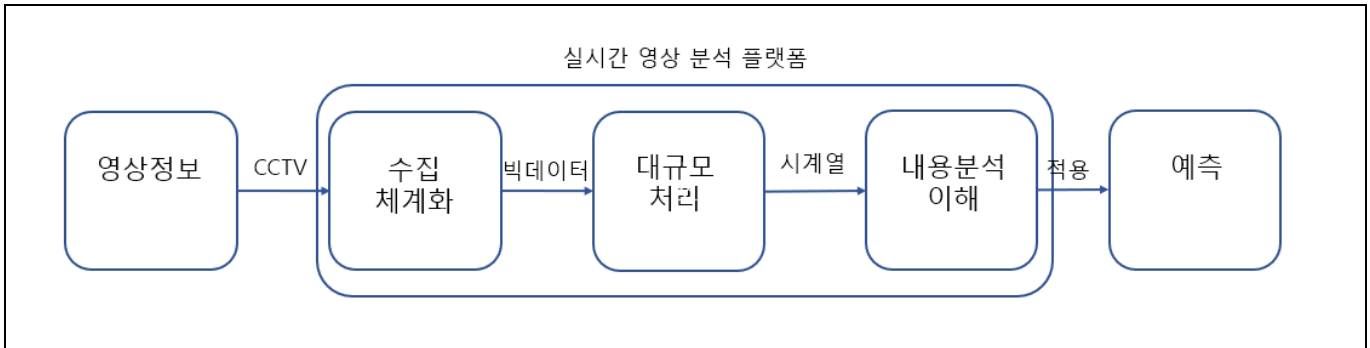
I. 인공지능 시각지능 딥뷰의 등장배경



- 인공지능 특이점 진화 과정 중 딥러닝, 기계학습, 지성/시각/음성 등의 다양한 인공지능 기술들이 필요하며 그 중 시각을 중심으로 한 인공지능인 딥뷰가 지속 활용되며 부각됨

II. 딥뷰(DeepView)의 개념도 및 설명

가. 딥뷰의 개념도



개념	- 사람이 사물을 인지하고 시공간적으로 상황을 파악하여 직관적으로 사물을 인지하듯 사람처럼 인지하는 시각 인공지능 기술

- 시각 인공지능 기술을 이용하여 다양한 사회 문제를 해결하기 위해 딥뷰를 사용

나. 딥뷰(DeepView)의 특징

특징	설명
영상 인식	- 영상 속 사람과 사물을 정확하게 인식
내용 파악	- 영상 속 인식 기반 내용이 무엇인지 사람만큼 정확하게 파악
실시간 분석	- 이미지, 동영상 수집으로 도심 재난, 범죄 위협 예측 실시간 수행 도와 줌
안정 정보 분석	- 대규모 시각 관련 빅데이터 구축으로 국가 차원 안정적인 정보 분석

- 기존의 범용 영상 데이터 활용과 관련하여 개발, 실제 행동 인식하는데 한계점을 딥뷰를 통하여 극복

III. 딥뷰(DeepView)의 기술요소

가. 데이터 수집 및 처리 기반의 기술요소

구분	기술요소	설명
시각정보수집	- 시각데이터 자산화	- 시각데이터 뱅크를 중심으로 실시간, 대규모 데이터 수집
	- 빅데이터 저장	- 비정형 대용량 데이터 저장기술
	- API 지원	- 시각 데이터 변환 및 외부서비스 I/F연계처리
대규모 처리	- 처리 파이프라인	- 대용량 비정형 시각데이터 처리방법 규격화 기술 병렬, 분산 처리 지원
	- 하이브리드 스케줄러	- 워크플로우 기반 분산처리와 GPU 사용 극대화 기술

- 데이터 수집, 처리에 이어 관련 내용 분석 및 예측 기반에 기술 필요

가. 내용분석 및 예측 기반의 기술요소

구분	기술요소	설명
내용분석	- 이미지 분석	- 객체와 움직임을 분석하고 상호관계 복합분석
	- 시각 텍소노미	- 이벤트, 객체에 대한 지식체계 구축
	- 시멘틱 추론	- 온톨리지 기반으로 복합동상 추론
예지형 응용	- 예측기술	- 시각정보 기반 재난/재해 등 선재적 예측
	- 실시간 대응	- 시각정보에 대하여 즉각적인 이벤트, 행위 발생 연계
지원	- 플랫폼 화	- 고성능 비쥬얼 디스커버리 플랫폼 지원

- 최근 ETRI에서 '실시간 대규모 영상 데이터 이해/예측을 위한 딥뷰 플랫폼 연구 개발 진행 중

IV. 딥뷰(DeepView)의 활용

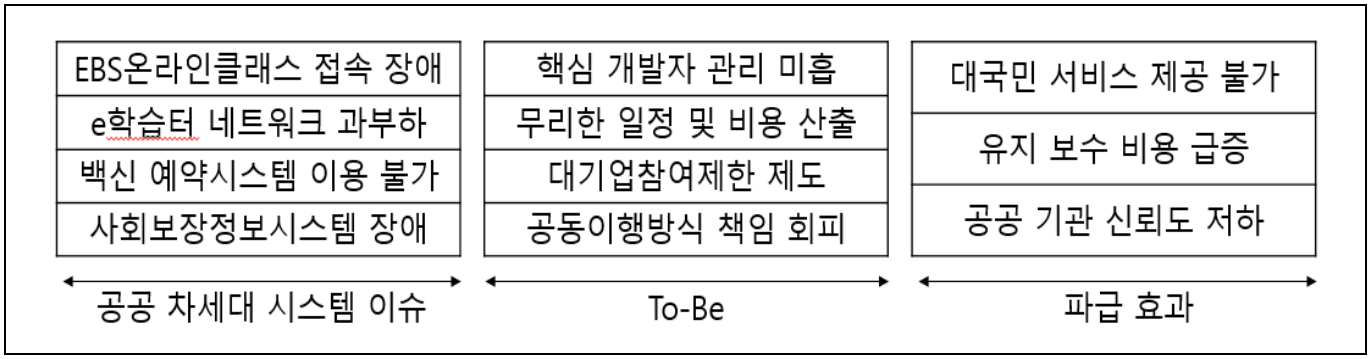
활용	설명
범죄, 사건 사전 감지	- CCTV 영상 기반으로 범죄, 사건의 사전 감지를 위한 활용
쓰레기 무단 투기 탐지	- 쓰레기 무단 투기 행위등과 같은 행위 사전 예방을 위한 활용
안전사고 예방	- 도심지에서 발생하는 안전사고 예방 및 신속 대응을 위해 활용

- 현재 딥 뷰(DeepView)는 다양한 사회적인 문제를 인공지능으로 해결하기 위한 활용 노력 중

"끝"

04	차세대 시스템 전환		
문제	최근 대규모 공공 차세대 시스템이 오픈 이후에 많은 문제점이 발생되어 사회적 불편을 초래하게 되었다. 이에 대하여 다음을 설명 하시오. 가. 발생된 문제점의 원인 나. 재발 방지를 위한 대책 및 법제도 보완 방안 다. 시스템 오픈 가능 여부 판단을 위한 지표 관리		
도메인	경영전략	난이도	중(상/중/하)
키워드	대기업참여제한제도, 정보시스템감리, 공동이행방식, 상생가산점		
출제배경	최근 진행한 공공 차세대 시스템 전환 이후 발생한 이슈 사항에 대한 논리적 대응력 확인		
참고문헌	22년도 국정감사 회의록		
해설자	서경석 기술사(제119회 정보관리기술사 / akslemf@naver.com)		

I. 시스템의 고도화와 통합화, 공공 차세대 시스템 오픈 관련 이슈



- 미흡한 사전 준비 및 관련 기업과 기관의 미진한 대응 등으로 인한 대국민 서비스 품질 및 신뢰도 저하

II. 발생된 문제점의 원인

가. 발생된 문제점의 기술적 원인

원인	설명
- 데이터 전환 미흡	- 기존 데이터 마이그레이션 후 정합성 확인 부족
- 도메인 분석 미흡	- 기존 시스템 통합에 따른 업무 영향도 및 범위 파악 부족
- 기능 미구현	- 공무원, 복지시설 근무자, 대국민 등 다양한 이용자의 필요 기능 불완전 구현
- 테스트 부족	- 시스템 통합 완료 후 미구현 기능, 일정 지연에 따른 운영 환경 전환 후 테스트

- 시스템 통합 시 필요한 사전 분석 및 충분한 테스트 미진행으로 인한 잠재적 문제점 발생 원인 발생

나. 발생된 문제점의 관리적 원인

원인	설명
- 일정 관리 미흡	- 일정의 지연이 있었으나 오픈 일정까지 미진한 부분 달성 가능할 것으로 오판
- 개발자 이탈	- 개발 핵심 인력 중 90% 이상이 과제 진행 중 이탈
- 안일한 대처	- 시스템 통합 완료 후 오류가 발생할 것이 예상되었으나 이후 대처 가능 판단
- 대기업 참여 제한	- 대기업 참여 제한에 따라 중소기업과 컨소시엄 구성 필수
- 의사 결정권	- 프로젝트 진행 시 T/F 내 의사 결정권 혼재에 따른 잠재적 위험 산재
- 상생 가산점	- SW산업진흥법 개정안에 따라 중소기업 비중 50% 이상 배정이 사실상 강제
- 공동이행방식	- 문제 발생 시 컨소시엄 지분이 많은 대표 사업자 책임 일임
- 컨소시엄 관리 미흡	- 서로의 시스템 구축 영역 명확히 구분되고, 서로의 영역에 관여할 수 없는 구조
- 감리 체계 미흡	- 감리 진행 시 산출물 위주의 감리 진행에 따른 실무적 확인 필요 부분 미진

- 기술적 원인보다 관리적 원인이 해당 문제 발생에 더 큰 영향이 있었다고 판단되어 관련 법제도 보완 필요

III. 재발 방지를 위한 대책 및 법제도 보완

가. 재발 방지를 위한 대책

원인	설명
- 사전 분석 철저	- 과제 진행 전 as-is 분석 강화를 통한 비즈니스 이슈 사전 확인
- PMO 활용	- 사업의 성공적 수행을 위한 기업 PMO, 공공 PMO 협업 진행
- 일정 관리 철저	- 일정 관리 철저화를 위한 WBS 기반의 일정 관리 및 상용 Tool 도입
- 원격지 개발	- 핵심 인력 이탈 방지를 위한 원격지 개발의 적극적 활용
- 전문가 활용	- 업무 도메인, 테스트, 데이터 등 필요 부분 전문가 활용 통한 잠재적 위험 제거
- 적정 수준 FP 산출	- 철저한 사전 분석 기반의 FP 도출을 통한 기능의 누락 예방 및 적정 비용 산출
- 데이터 품질 검증	- 데이터 마이그레이션 후 정합성, 무결성 확보를 위한 데이터 품질 검증 이행
- 결정권 일원화	- 중요 사항 결정 시 혼선 예방을 위한 과제 내 T/F의 결정권자 관리
- 강소 기업 육성	- 컨소시엄의 효율적 구성을 위한 정부 주도 중소 기업 육성 진행

- 사업 진행 시 기업과 기관의 인식 개선을 통한 위험 요소 사전 제거 대책 마련 필요

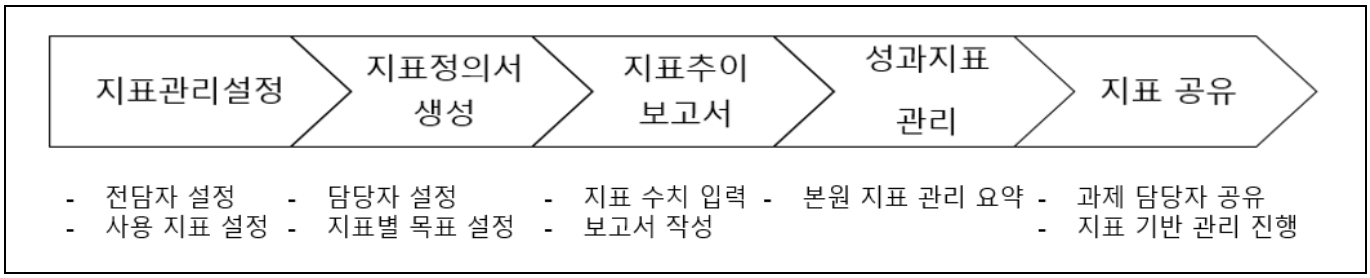
나. 재발 방지를 위한 법제도 보완

원인	설명
- 대기업참여제한 제도 완화	- 대기업 참여 가능 사업 규모 완화를 통한 대국민 서비스 안정성 확보
- 분리/분할 발주 강화	- 사업 규모에 따라 적절한 분리, 분할 발주를 통한 과제 위험 완화
- 분담 이행제	- 사업 참여자 모두에게 책임을 분담함으로써 업체의 적극적 관여 추진
- 컨소시엄 구성 체계 변경	- 동일 사업 참여 컨소시엄의 책임 및 권한 범위 조정
- 감리 강화	- 감리 시 업무 부문 감리 항목 강화 및 감리 완료 후 사후 관리 강화
- 상생가산점 개정	- 상생가산점 산정 시 중소기업 참여 비율 조정 통한 사업자 선정 최적화
- 발주 기준 상세화	- 공공 사업 발주 시 발주 기관의 지표 항목 개선 통한 사전 위협 제거
- 하도급 제도 개선	- 컨소시엄 구성 시 하도급 기업 관리 기준 개정 통한 관리 효율화
- SW산업진흥법 개정	- 공공SW사업 진행 시 기본이 되는 SW산업진흥법 개정 통한 사업 현실화

- 기업의 인식 개선만으로는 한계가 있으므로 법제도의 보완을 통한 사업 참여자의 책임 의무 강화

IV. 시스템 오픈 가능 여부 판단을 위한 지표 관리

가. 시스템 오픈 가능 여부 판단을 위한 지표 관리 절차



- 시스템 오픈 전 가능 여부 판단 지표 관리 프로세스화를 통한 시스템 오픈 후 안정적 운영 추구

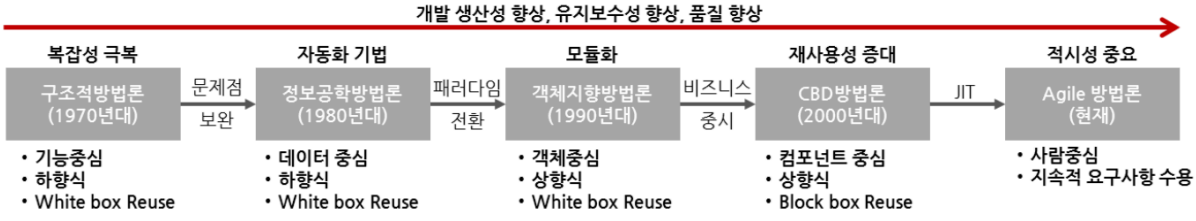
나. 시스템 오픈 가능 여부 판단을 위한 지표

구분	세부	설명
정합성	- 데이터	- 시스템의 기본적 서비스 제공을 위한 데이터의 정합성 확인
	- 기능	- 설계 시 도출된 사용자 요구 기능 사항 충족 여부 확인
	- 산출물	- 사업계획서에 명시된 산출물의 정상 산출 여부 확인
	- 업무 연계	- 타 시스템과의 업무 상 연계 오류 및 지연 발생 여부 확인 필요
기술	- 테스트	- 블랙박스, 화이트박스, V 모델 테스트 통한 테스트 완전성 확보
	- 통합 I/F	- 시스템 통합 발생 시 I/F의 정상적 수행 여부 확인 필요
	- 품질	- 품질 요구 사항 확인을 위한 품질 항목 점검 진행
	- 보안성	- 개인정보, 시스템 중요 정보에 대한 보안성 확보 여부 확인
비기능	- 성능	- 부하/데이터 입출력 테스트 통한 시스템 사용자 체감 성능 달성
	- 호환성	- 다양한 언어 기반 시스템의 상호 호환성 확보 여부 확인
	- 내고장성	- 장애 상황에도 시스템이 적절한 수준의 복구 및 정상 동작 여부
	- 사용성	- 시스템 실제 사용자의 시스템 사용상 불편 여부 확인

"끝"

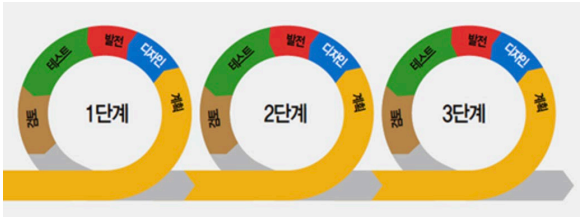
05	Agile 방법론		
문제	A 기업은 다수의 기존 정보시스템을 운영 및 유지보수를 하고 있으며 신규 시스템에 대한 개발을 기획 중에 있다. 개발 방법론으로 구조적 방법론을 주로 활용하여 왔지만 Agile 방법론의 도입을 검토하고 있다. 다음의 사항에 대하여 설명하시오. 가. 구조적 방법론과 Agile 방법론 비교 나. Agile 방법론의 스크럼(Scrum)과 칸반(Kanban)설명 다. Agile 방법론의 효율적인 수행 방안 제시		
도메인	소프트웨어공학	난이도	하(상/중/하)
키워드	절차, 의사소통, Sprint, WIP		
출제배경	애자일 방법론의 기본 지식 확인		
참고문헌	Research And Improve Agile Development Methods Based On Teamwork yungki Kim		
해설자	이상용 기술사(제 124회 정보관리기술사 / orangeday77@gmail.com)		

I. 개발방법론 개요

구분	설명
개념	- 소프트웨어를 개발하는 방법에 대한 이론으로서, 소프트웨어 개발 과정, 절차, 방법, 산출물, 기법, 도구들을 체계적으로 정리하고 표준화시킨 방법론
구성 요소	작업절차 → 소프트웨어를 진행할 때 이루어지는 작업의 순서 작업방법 → 각 단계별 작업마다 수행해야 할 일(누가, 언제, 무엇을) 산출물 → 단계별로 나오는 산출물(설계서, 명세서) 관리 → 개발 진행을 어떻게 제어하고 감독할 것인지 기법 → 단계별 작업 시 사용하는 기술, 기법(DFD, ERD, Use Case) 도구 → 사용하는 기법 별 지원 도구(파워포인트, 엑셀, ERWin)
주요 유형	

- 소프트웨어의 위기 극복방안으로 개발프로세스 모델과 다양한 개발방법론 제시

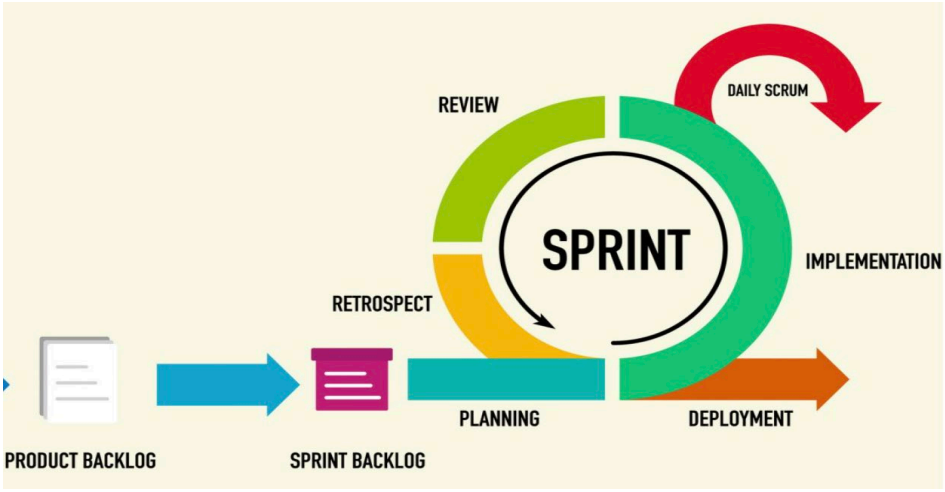
II. 구조적 방법론과 Agile 방법론 비교

구분	구조적 방법론	Agile 방법론
개념	- 절차 중심의 구조중심 분석/설계 그리고 문서화 모듈화 중심의 소프트웨어 개발 방법론	- 상호협력 및 내/외부 변화에 대한 민첩한 대응을 추구하는, 반복점진적 소프트웨어 개발 방법론
개념도		
특징	- 요구사항 분석및 요구사항 추출 기법 활용 - 기능 중심, 모듈의 응집도와 결합도 고려 - Top-Down 방식의 업무활동 구체화	- 초기테스트, 의사소통, 상호작용 중시 - 고객과 개발팀의 소통중심 분석&설계 - Bottom-Up 방식의 SW구체화
관점	- 기능 중심 - 분할과 정복	- 고객과의 지속적인 협력 - 작동하는 소프트웨어 구현
기법	- 정형화 - 모듈화 - 사용자 요구사항 문서화	- XP - SCRUM - KANBAN - LEAN
도구	- Data Flow Diagram - Data Dictionary - State Transition Diagram	- CI/CD - 테스트 자동화, 커버리지 도구 - TDD를 위한 테스트 구현 도구
프로세스	- 요구사항 분석 - 구조적 분석 - 구조적 설계 - 구조적 프로그래밍	- Plan - Design - Development - Test & Feedback & Iteration
산출물	- 요구사항정의서 - DB설계서	- 산출물 작성 수준 최소화 - 엔지니어링 도구 기반한 산출물 생성 자동화

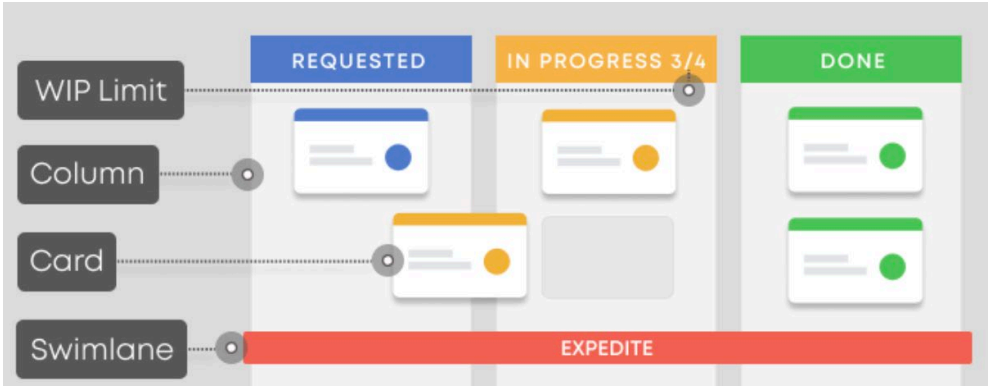
- Agile 방법론을 실현하는 기법 중 스프린트 기반의 SCRUM과 Work In Process 기반의 KANBAN이 있음

III. Agile 방법론의 스크럼(Scrum)과 칸반(Kanban)설명

가. Agile 방법론의 스크럼(Scrum)설명

구분	설명
개념	- Product Backlog를 Sprint Backlog로 분할하여 주기적으로 Release를 수행하는 Agile 기반의 개발방법론
개념도	
원리	- Sprint를 통한 반복적 개발
구성요소	- 주체(Product Owner/Scrum Master/Team), 미팅(Sprint Planning/Daily/Sprint Review)
단위	- Sprint backlog를 통한 time boxing
관리	- Burn down chart와 velocity 측정

나. Agile 방법론의 칸반(Kanban)설명

구분	설명
개념	- 칸반보드를 통해 개발 공정을 시각화 하고 WIP제한을 이용하여 workflow 상의 공정을 관리 및 최적화 하는 Lean기반의 개발방법론
개념도	
원리	- Workflow를 통한 연속적 개발
구성요소	- WIP Limit, Column, Card, Swimlane
단위	- WIP제한을 통한 작업량 조절
관리	- Workflow 가시화와 WIP 제한

IV. Agile 방법론의 효율적인 수행 방안 제시

가. 프로세스 관점에서의 Agile 방법론의 효율적인 수행 방안 제시

구분	핵심 기술
사전준비	- 프로세스 정립 위한 애자일 가이드라인 배포 - 경영진 참여 유도 / 이해관계자 식별
요구정의	- Product backlog / Sprint backlog 작성 및 문서화 - 요구사항의 Story화
분석/설계	- 의사소통 채널 정리 - 분석/설계 반복 수행
개발/검증	- 작업과 개발자의 분배문제 극복하기위한 팀워크 활동 - 개발 반복 수행 - TDD 적용
배포	- 정기 배포 자동화 - CI/CD 프로세스 도입
회고	- 기술 부채 정리 - 회고 미팅 후 다음 스프린트 이관

나. 개발과 조직 관점에서의 Agile 방법론의 효율적인 수행 방안 제시

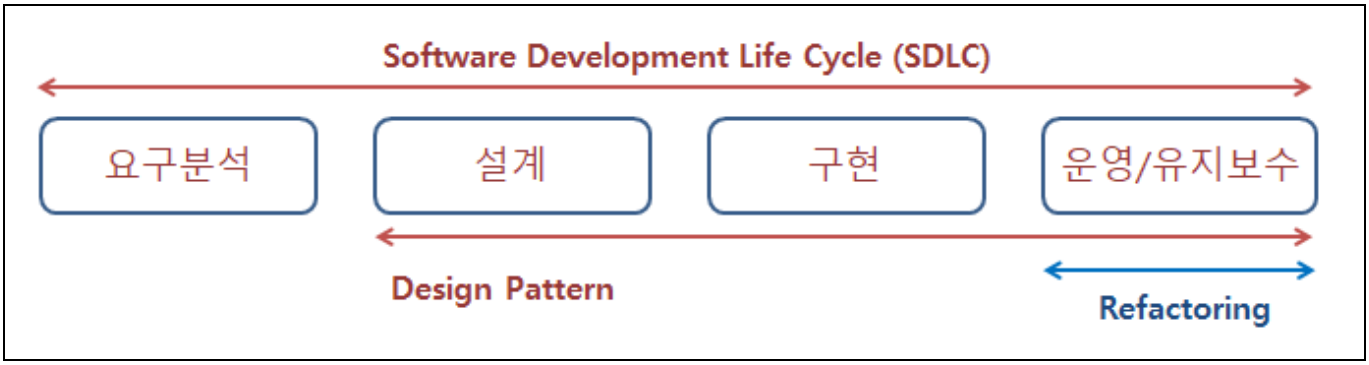
구분	핵심 기술
준비	- 작업 요구 능력 및 팀원 능력 수집
측정	- 팀원 능력 측정
우선순위	- 작업 난이도와 순위 결정
배치	- 작업에 개발인력 적절 배치
검증	- 인력 배치 적절성 검증
문서화	- 프로젝트 문서화

- 전통적 방법론과 애자일을 혼합한 하이브리드 방식의 개발 방법론 도입 사례 증가

"끝"

06	리팩토링, 디자인 패턴		
문제	객체지향 기법 중에는 리팩토링(Refactoring)과 디자인패턴(Design Pattern)이 있다. 두 기법을 각각 정의하고 공통점과 차이점에 대하여 설명하시오.		
도메인	SW공학	난이도	중(상/중/하)
키워드	디자인 패턴(생성, 구조, 행위), 리팩토링(응집도, 결합도, 가독성)		
출제배경	SW 개발부터 유지보수까지 디자인패턴과 리팩토링의 적용 범위 및 활용에 대한 개념 학습		
참고문헌	ITPE 기술사회 자료, 한국정보처리학회 2002년도 추계학술발표논문집		
해설자	장건환 기술사(제 126회 정보관리기술사 / jkh556@naver.com)		

I. SW의 효율적인 설계와 유지보수를 위한 디자인패턴과 리팩토링 개요



- SW의 효율적인 개발과 유지보수를 위해 설계, 구현 및 유지보수 전반에 적용하는 디자인패턴과 리팩토링
- SW개발 설계에 따라 유연한 리팩토링이 가능하므로 디자인패턴과 리팩토링은 SW개발 전반에 활용

II. 디자인패턴과 리팩토링의 정의

가. 디자인패턴 정의 및 형식

구분	설명	
정의	- SW 디자인에서 통속적으로 발생하는 문제에 대한 일반적으로 재사용이 가능한 해결책 혹은 경험적인 솔루션 - 특정 영역의 설계 문제를 해결하기 위해 고안된 형식적인 방법	
디자인패턴 형식	이름(Name)	- 패턴 자체의 내용을 효과적으로 전달 할 수 있는 이름
	종류 (Classification)	- 여러 개의 패턴을 체계적으로 분류 - 생성(Creational) : 객체들의 생성과 관련 - 구조(Structural) : 클래스, 객체의 정적인 구조와 관련 - 행위(Behavioral) : 클래스와 객체의 반응과 책임할당
	의도 (Intent)	- 이 패턴이 무엇을 하며, 어떤 의도로 작성되었는지 무엇을 해결하는지를 설명하여 기술
	별칭 (Also Known as)	- 위의 공식적인 이름 외에 잘 알려진 다른 이름
	동기 (Motivation)	- 이 패턴이 해결하여야 하는 디자인 문제와 그것을 해결하기 위한 클래스와 객체들이 어떻게 사용되는지에 대해 시나리오 형식으로 기술

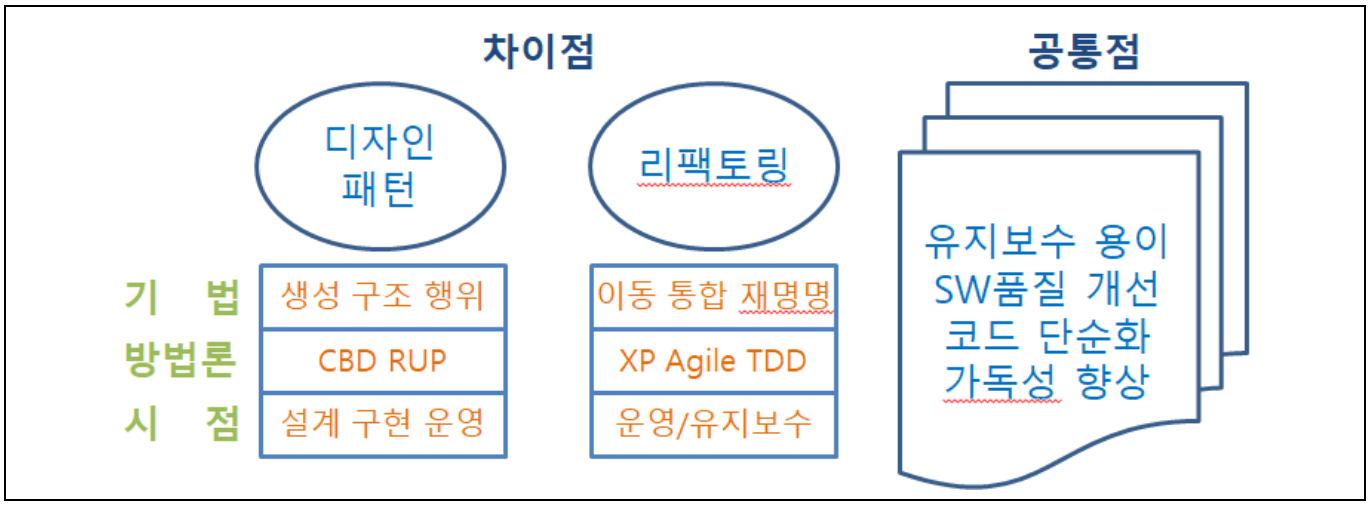
	구조 (Structure)	- 패턴 안에서 문제를 해결하기 위해 사용되는 클래스와 객체의 구조 UML 다이어그램을 통해 표현
	구성물	- 구조항목에 포함된 각종 클래스, 객체의 의미와 그 책임을 설명
	협력과정 (Collaboration)	- 각 클래스와 객체가 자신에게 맡겨진 책임을 수행하기 위해 서로 메시지를 주고 받는 과정을 묘사
	결과 (Consequence)	- 패턴이 목적을 달성하기 위해 어떤 면을 해결하는지 설명하고 패턴을 적용할 때 발생할 수 있는 문제점과 패턴 적용시 효과 시스템 상황에 대해 변동하는 부분들을 기술
	샘플코드	- 특정언어로 패턴을 구현한 예제
	관련 패턴	- 본 패턴과 유사하거나 밀접하게 관련된 다른 패턴

나. 리팩토링의 정의 및 주요기법

구분	설명		
정의	- SW의 이해성 및 유지보수성을 높이기 위해 Code Smell을 제거하여 동작의 변화없이 내부구조를 변경하는 활동 - 결과의 변경없이 코드의 구조를 재조정하여 가독성을 높이고 유지보수를 편하게 하는 유지보수 활동		
주요 기법	결합도 측면	Move Method	- Method가 자신의 정의된 클래스보다 다른 클래스에서 더 많이 사용하고 있으면 Move해야함
		Move Attribute	- 타 클래스와 결합 높은 속성을 이전
		Extract Class	- 두 개의 클래스가 해야 할 일을 하나의 클래스가 하고 있다면 새로운 클래스를 만들어 관련 있는 필드와 메소드를 이전
	응집도 측면	Push Down Method	- 서브 클래스만 사용하는 메소드 이전
		Push Down Attribute	- 서브 클래스만 사용하는 속성 이전
		Inline Class	- 별 기능 없는 Class를 다른 것과 합침
	일반화	Full up Field	- 두 서브 클래스가 동일한 필드를 가지고 있다면 그 필드를 슈퍼 클래스로 올림
		Full up Method	- 동일한 기능을 하는 메소드를 여러 서브 클래스에서 가지고 있다면 수퍼클래스로 올림
	단순화	Replace temp w/Query	- 어떤 수석의 결과 값을 저장하기 위해 임시 변수를 사용한다면 수식을 Method로 만들고 임시 변수를 참조하는 곳을 모두 Method 호출로 수정
		Rename Mehod	- Method Name이 그 목적에 맞게 이름을 변경
	은닉	Symbolic Constant	- Literal 숫자를 의미 있는 이중의 Constant로 바꿈
		Encapsulate Field	- Public 필드가 있는 경우 그 필드를 Private으로 만들고 접근자를 제공

III. 디자인 패턴과 리팩토링의 공통점과 차이점

가. 디자인 패턴과 리팩토링의 공통점과 차이점 개념도



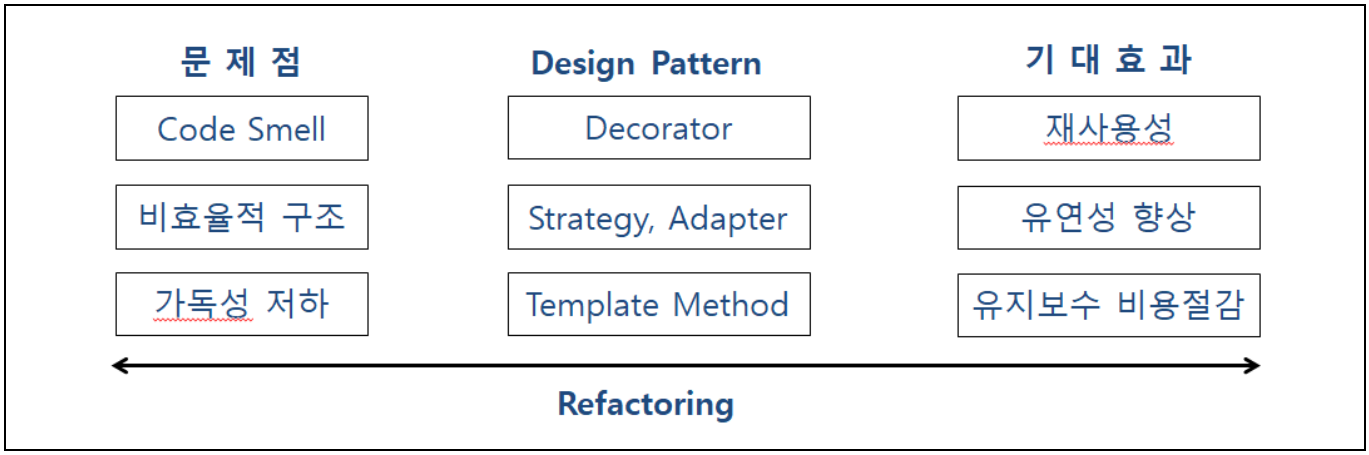
나. 디자인 패턴과 리팩토링의 공통점

구분	디자인 패턴	리팩토링
목적	- App의 구조를 개선하고 단순화하여 개발/운영을 용이하게 하기 위함	
품질	- SW 품질 보증, 표준 기반하여 능률 향상	
만족도	- 사용자 요구 만족, 서비스 시간 단축, 기능의 유연성 향상	
성능 개선	- 효율적인 구조로 애플리케이션 성능 향상	
유지보수 용이	- 구조의 단순화와 표준화된 방식을 활용하여 유지보수성 향상	
생산성 향상	- Clean Code와 재사용성을 높여 생산성을 향상	
쉽게 이해	- 가독성 있는 Coding으로 개발자가 이해하기 용이	
안정성	- SW의 SDLC 주기가 늘어나면서도 안정적으로 유지보수 가능	

다. 디자인 패턴과 리팩토링의 차이점

구분	디자인 패턴	리팩토링
시점	- 설계, 구현, 운영/유지보수	- 운영/유지보수
범위	- 반복되는 문제점의 해결	- 소스코드 정제
주체	- 설계자, 개발자	- 운영자, 개발자
기법	- 생성, 구조, 행위	- Move, Extract, Inline, Rename
방법론	- CBD, RUP	- XP, Agile, TDD
효과	- 기능에 대한 변동여부 보다 더 효율적인 구조 개선이 초점	- 드러나는 기능 변동 없이 성능이 개선
대상	- 객체지향 설계 위한 SW - 통속적으로 발생되는 문제	- 중복된 코드, 거대한 클래스, 변경의 분산
관점	- 개발자 관점에서 경험 기반으로 모델링	- 사용자 관점에서 변화 없이 성능 품질 향상

IV. 디자인패턴을 활용한 리팩토링



- 문제점을 도출하고 Design Pattern을 통해 리팩토링을 수행하여 운영/유지보수에 용이하도록 개선
- Decorator, Strategy & Adapter, Template Method 패턴 등을 이용하여 재사용성과 유연성을 향상

"끝"

ITPE 기술사회
제129회 정보처리기술사 기출문제 해설집

대 상 정보관리기술사, 컴퓨터시스템응용기술사, 정보통신기술사, 정보시스템감리사 시험
발행일 2023년 02월 04일
집 필 강정배PE, 소민호PE, 전일PE, 정상PE, 석PE, 이상용PE, 장건환PE
출 판 **ITPE(Information Technology Professional Engineer)**
주 소 ITPE 대치점 서울시 강남구 선릉로 86길 17 선릉엠티빌딩 7층
ITPE 선릉점 서울시 강남구 선릉로 86길 15, 3층 IT교육센터 아이티피이
ITPE 강남점 서울시 강남구 테헤란로 52길 21 파라다이스벤처타워 3층 303호
ITPE 영등포점 서울시 영등포구 당산동2가 하나비즈타워 7층 ITPE
연락처 070-4077-1267 / itpe@itpe.co.kr