

ICT의 가치를 이끄는 사람들!!!

127회

정보관리기술사 기출풀이 3교시

국가기술자격 기술사 시험문제

정보처리기술사 제 127 회

제 3 교시

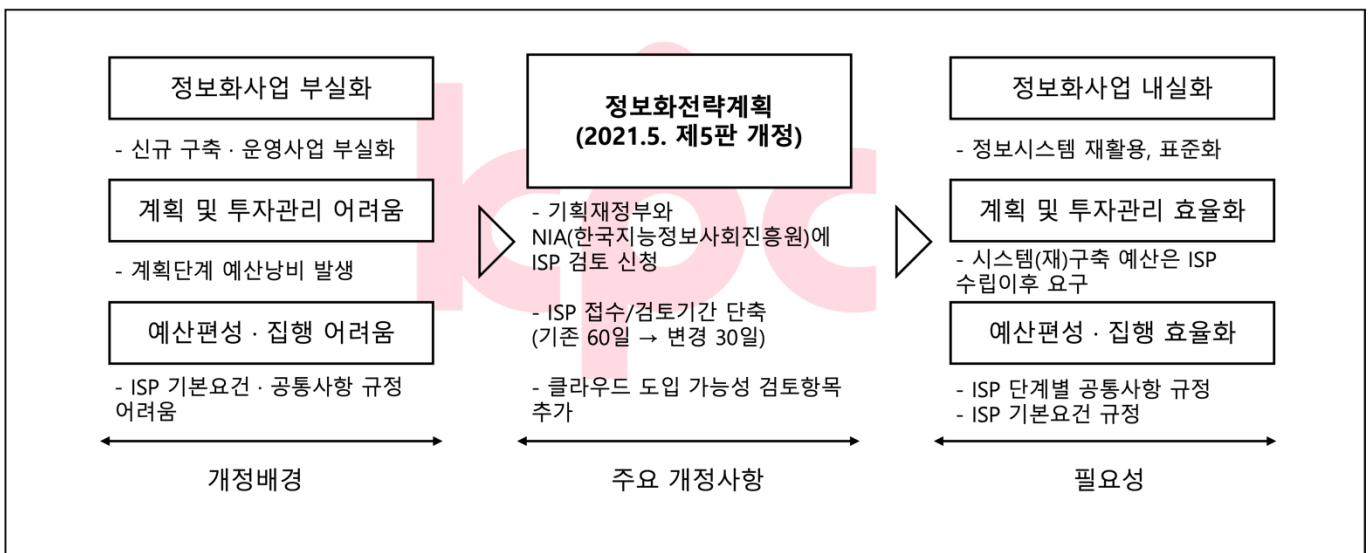
분야	정보처리	종목	정보관리기술사	수험 번호	성명
----	------	----	---------	----------	----

※ 다음 문제 중 4 문제를 선택하여 설명하시오. (각 25 점)

1. 정보전략계획(Information Strategy Planning)에 대하여 아래의 사항을 설명하시오.
가. 단계별 활동 및 산출물
나. ISMP(Information System Master Plan)와 비교
2. 리팩토링(Refactoring)에 대한 아래의 사항을 설명하시오.
가. 정의, 목적, 리팩토링 순서, 리팩토링 주요기법
나. 코드스멜(Code Smell)의 정의와 특징
다. 코드스멜의 종류를 3 개 이상 기술하고 각각의 리팩토링 방법
3. A 기업에서는 비즈니스 수행과정에서 수집된 많은 양의 빅데이터(Bigdata)를 통합 관리하고자 한다. 데이터 관리에 대한 아래의 사항을 설명하시오.
가. 데이터 거버넌스(Data Governance)의 개념 및 주요 기능
나. 마스터 데이터(Master Data)의 개념과 필요성
다. 마스터 데이터 관리(Master Data Management)의 구성요소와 구축 시 고려사항
4. 데이터베이스 옵티마이저(Optimizer)에 대한 아래의 사항을 설명하시오.
가. 옵티마이저의 개념
나. RBO(Rule Based Optimizer)와 CBO(Cost Based Optimizer) 비교
다. 옵티마이저의 적용 시 고려사항
5. 소프트웨어 정의 네트워크(SDN)에 대한 아래의 사항을 설명하시오.
가. SDN 제어평면의 개요 및 구조의 특징
나. 오픈플로우(OpenFlow) 프로토콜
6. SOAR(Security Orchestration, Automation and Response)의 개념 및 등장배경, 구성 요소, 주요 기능, 기대효과, 도입 시 고려사항에 대하여 설명하시오.

문 제	1. 정보화전략계획(Information Strategy Planning)에 대하여 아래의 사항을 설명하시오.		
	가. 단계별 활동 및 산출물 나. ISMP(Information System Master Plan)와 비교		
출 제 영 역	SW	난 이 도	★★☆☆☆
출 제 배 경	- ISP 수립 공통가이드 개정(5 판), 민간투자형 SW 사업 활성화		
출 제 빈 도	관리 123 회 2 교시, 관리 119 회 4 교시, 모의고사/합숙 다수 출제		
참 고 자 료	- 정보화전략계획 수립 공통가이드 5 판 – NIA 한국지능정보사회진흥원 - 전자정부지원사업 정보화전략계획(ISP) 산출물 점검 가이드 – 행정안전부 - 정보시스템 마스터플랜(ISMP) 방법론 – 정보통신산업진흥원		
Key word	- ISP(전략수립, 30 일, 클라우드), ISMP(상세분석, RFP, FP)		
풀 이	이정현(125 회 정보관리기술사)		

1. 정보화전략계획(ISP) 개요.



- ISP 는 EA, ISP, ISMP 순으로 수립되며, 조직의 경영 목표 전략을 효과적으로 지원하기 위한 IT 사업(과제)를 도출하는 수립 활동임.
- 단계별 활동은 환경분석, 현황분석, 정보화비전 및 전략수립, 목표모델설계, 통합이행계획으로 진행되며, 각 활동별로 산출물이 존재.

2. 정보화전략계획(ISP)의 단계별 활동 및 산출물

1) 정보화전략계획(ISP)의 단계별 활동

단계	활동	설명
환경분석	- 경영환경분석	- 외부환경 요인과 경영전략 분석 통해 대응방안 도출
	- 법령·제도 분석	- 사업에 영향을 미칠 수 있는 요구사항 도출
	- IT 환경 분석	- 최신 정보기술 및 기술환경 변화 검토

현황분석 (As-Is 분석)	- 업무현황 분석	- 조직역할 및 업무체계 분석 후 현업무상의 요구 도출
	- IT 현황 분석	- 업무시스템, 데이터, 인프라, IT 거버넌스 분석
	- 벤치마킹	- 현황분석에서 도출된 요구사항 기반으로 벤치마킹 대상 선정 및 선진사례 조사·분석을 진행
	- GAP 분석	- 도출된 정보화 요건과 현 정보화요건 차이분석
	- 이슈통합 및 개선과제 도출	- 연관성 높은 이슈사항을 그룹화 및 근본원인 분석 후 개선과제 도출
정보화 비전 및 전략 수립	- 정보화 비전	- 환경분석과 현황분석 결과를 연계하여 정보화 비전, 목표, 단계별 실행 전략 수립
	- 정보화 전략 수립	- 정보시스템 구축 원칙, 적용기술요건, 전략 수립
목표모델설계 (To-Be Model)	- 개선과제 상세화	- 현황분석에서 도출된 개선과제 상세화
	- 업무프로세스 설계	- 개선과제, 선진사례를 고려하여 업무프로세스 최적화
	- 구조 설계	- 정보시스템, 데이터구조, 기술 및 보안구조 설계
통합 이행계획	- 통합 이행계획 수립	- 개선과제간의 선후관계를 고려하여 실행일정 수립
	- 총구축비 산출	- SW 개발비의 이행과제별 FP 도출 및 장비비 도출
	- 효과분석	- 타당한 기대효과 분석(B/C, NPV)

- 정보화전략계획 단계별 산출물은 개발방법론에 따라 순서 및 세부내용은 탄력적으로 조정 가능.

2) 정보화전략계획(ISP)의 산출물

단계	산출물	설명
환경분석	- 경영환경분석서	- PEST, ETRIP, STEEP 기반 거시적 환경 분석 내용 기재
	- 법령·제도 분석서	- 사업에 영향을 미칠 수 있는 법 제도 도출
	- 정보기술 동향 분석서	- 주요정보기술 동향 및 정보기술 적용성 분석
현황분석 (As-Is 분석)	- 업무현황분석서	- 현행 프로세스 맵, 프로세스 기술서, 업무상관도
	- 선진사례 동향 파악서	- 비즈니스 및 정보시스템 관점에서 선진사례 파악
	- 차이 분석서	- 현재 IT와 목표 IT 간 희망 성과 및 실제 성과 비교
	- 요구사항·개선과제 분석서	- 요구사항정의서, 개선과제 요약서 도출
정보화 비전 및 전략 수립	- 정보화 전략 정의서	- 비전, 목표, 과제, 추진 단계별 전략, 기술요건 기재
목표모델설계 (To-Be Model)	- 과제 상세정의서	- 미래 프로세스 맵, 프로세스 체계도, 상세 목표
	- 업무프로세스 설계서	- 프로세스 개선 전략, 프로세스 개선 일정 수립
	- 정보시스템 구조 설계서	- 응용 구조 정의서, 통합정보시스템 기술구조도
	- 데이터 구조 설계서	- 개념/논리/물리 데이터 설계 및 테이블 정의서(ERD)
	- 기술 및 보안 구조 설계서	- 보안정책, 보안구성도 및 정의서, 보안관계기술서
통합 이행계획	- 통합 이행계획 수립서	- 정보시스템 구축일정 전환 전략, 투자효과분석

- ISP 이후 과제가 도출되고, ISMP를 통해 정보시스템의 기능·기술적 요구사항 상세화.

- ISP가 거시적(IT 비전 및 전략) 측면이라면 ISMP는 근시적(FP, HW 비용)측면 차이가 존재.

3. 정보화전략계획(ISP)과 정보시스템 마스터플랜(ISMP)의 비교

1) 정보화전략계획(ISP)과 정보시스템 마스터플랜(ISMP)의 비교

비교항목	정보화전략계획(ISP)	정보시스템 마스터플랜(ISMP)
목적	<ul style="list-style-type: none"> - 경영전략과 정보화 전략 연계 - 정보시스템 신기술 반영 - 거시적 관점, 전략정의서 작성 	<ul style="list-style-type: none"> - 정보시스템 기능·기술 요구사항 상세화 - Function Point 가 도출될때까지 상세화 - 근시적 관점, RFP 작성
범위	<ul style="list-style-type: none"> - 전사, 서비스, 부서 대상 정보화 전략 	<ul style="list-style-type: none"> - 단위 프로젝트, 단위 프로젝트 묶음
주요 활동	<ul style="list-style-type: none"> - 경영환경 분석 - 내부환경 분석 - 정보화 비전 및 목표 수립 - 정보화 전략수립 - 정보구조 설계 - 투자 계획 및 예산 수립 	<ul style="list-style-type: none"> - 구축범위 및 방향성 수립 - 기능 및 기술요건 정의 - 요건 상세기술 - 추진 이행계획 - 수행조직 정의 - 예산선정, 업체선정

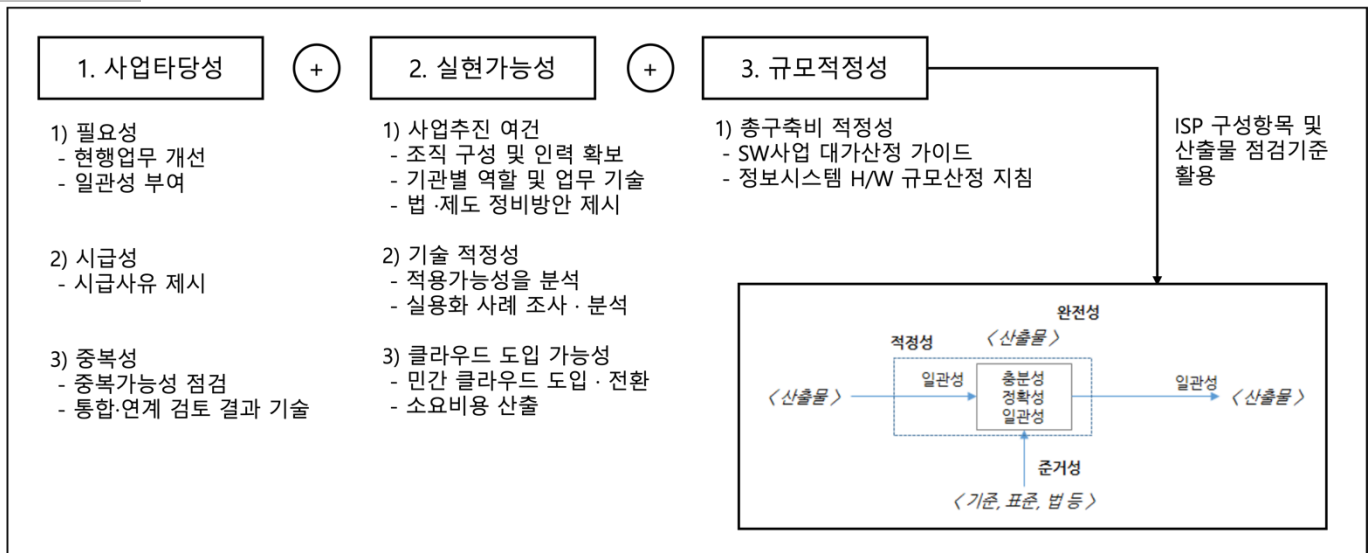
- 정보화전략계획(ISP)과 정보시스템 마스터플랜(ISMP)은 특히 프로세스 및 산출물의 차이가 존재

2) 정보화전략계획(ISP)과 정보시스템 마스터플랜(ISMP)의 상세비교

비교항목	정보화전략계획(ISP)	정보시스템 마스터플랜(ISMP)
프로세스	<ul style="list-style-type: none"> - 환경분석 - 현황분석 - 정보화 비전 및 전략 수립 - 목표모델 설계 - 통합이행계획 	<ul style="list-style-type: none"> - 프로젝트 착수 및 참여자 결정 - 정보시스템 방향성 수립 - 비즈니스 및 정보기술 요건 분석 - 정보시스템 구조 및 요건 정의 - 정보시스템 구축사업 이행방안 수립
주요 산출물	<ul style="list-style-type: none"> - 경영환경 분석서 - 기술동향 분석 보고서 - 차이 분석서 - 업무/정보/데이터/보안 설계서 	<ul style="list-style-type: none"> - 상세요건 정의서 - RFP(제안요청서) - 정보시스템 예산(FP, HW 규모산정) - 업체 평가항목 기준서
기대효과	<ul style="list-style-type: none"> - 중장기적 업무 전략 - 최신 정보 기술 도입 전략 방향 제시 - 정보화 예산 효율화 및 내실화 목적 	<ul style="list-style-type: none"> - RFP의 사업범위, 사업목표 구체적 작성으로 예산 절감 가능 - 기업 업무 범위 및 투자예산 산정 통한 고객 요구사항 변경 대응

- ISP 수행초기부터 주요구성항목과 산출물 점검기준을 고려할 경우, 추후 정보시스템 예산 절감에 효과증가.

4. 성공적인 정보화전략계획을 위한 주요구성항목 및 산출물 점검기준



- 초기 ISP 수립시 구성항목 및 산출물 점검기준을 활용하여, 과제 구체화시 예산절감에 탁월한 효과 기대. "끝"

기출풀이 의견

1. EA > ISP > ISMP 흐름을 가져가시면 논리적 흐름이 가시화되기 때문에, 친절한 답안을 가져갈 수 있었던 문제였습니다. 또한 과거 기출문제, 모의고사에서도 많은 출제가 되었던 토픽이며, 작성대비 ROI가 적은 토픽으로 볼 수 있으므로, 대응용으로 준비하시면 좋겠습니다.

문 제	2. 리팩토링(Refactoring)에 대한 아래의 사항을 설명하시오. 가. 정의, 목적, 리팩토링 순서, 리팩토링 주요기법 나. 코드스멜(Code Smell)의 정의와 특징 다. 코드스멜의 종류를 3 개 이상 기술하고 각각의 리팩토링 방법		
출 제 영 역	SW	난 이 도	★★☆☆☆
출 제 배 경	- SW 중심사회 전환, 클라우드 워크로드, 기술부채방지		
출 제 빈 도	관리 119 회 4 교시, 관리 101 회 3 교시, 모의고사 다수 출제		
참 고 자 료	<ul style="list-style-type: none"> - 리팩토링, Refactoring 이란? – 이끼의 생각 - Code Refactoring – devopedia - 소프트웨어 리팩토링 (Refactoring) – 도리의 디지털라이프 - 리팩토링(Refactoring) – codereviewers 		
Key word	- 리팩토링(기능변경없이, 품질향상, 이동, 추출, 변경), 코드스멜(동작가능, 중복, 주석, 긴메소드)		
풀 이	이정현(125 회 정보관리기술사)		

1. 리팩토링의 정의, 목적, 순서, 주요기법

1) 리팩토링의 정의, 목적, 순서

개념	- 기능 변경 없이 Code Smell 제거, 가독성/로직개선/성능 향상 통한 SW 안정성/신뢰성/재사용성 향상 기법	
목적	- 파일의 무결성, 실행권한 등에 관한 유효성 검증방법 및 부적합한 파일에 대한 처리방법을 설계하는 개발보안활동	
순서	1) 리팩토링 코드 작성 2) 테스트 실행 3) 성공시 리팩토링 대상 선정 4) 실패시 리팩토링 코드 수정 5) 회귀테스트 반복	<pre> graph LR Developer[Developer] -- Make change --> RefactoredCode[Refactored code] RefactoredCode -- Run tests --> TestsFail[Tests fail] RefactoredCode -- Run tests --> TestsPass[Tests pass] TestsFail -- Revert refactoring --> Developer TestsPass -- Decide on next refactoring --> Developer </pre>

- 리팩토링은 품질향상에 중요하며, 방법론적 측면과 코드기반 측면으로 분류가능.

2) 리팩토링의 주요기법

구분	주요기법	설명
방법론 측면	- TDD Refactoring	- TDD 개발 주도 방법론을 이용한 기법
	- Litter-Pickup Refactoring	- 문제발생시 즉시 코드를 정리, 수정하는 기법
	- Preparatory Refactoring	- 이해중심기반으로 작업시작시 리팩토링을 통해 기술부채를 줄이는 기법
	- Planned Refactoring	- 시스템 통합 및 개선범위가 큰 경우 수행하고, 리팩토링 스토리를 이용한 기법

	- Long-Term Refactoring	- 장기적으로 현재기능과 추후 도입될 기능을 고려하여 추상화와 분기를 이용한 기법
코드측면 주요기법	- Move Method	- 서브클래스 동일필드 보유 시 슈퍼클래스로 이동
	- Full up Field	- 정의된 클래스보다 타 클래스 다수 호출 시 이동
	- Full up Method	- 동일 기능 메서드를 슈퍼 클래스로 이동
	- Extract Method	- 그룹으로 묶을 수 있는 코드의 메서드
	- Extract Class	- 두 개의 클래스 업무를 신규 클래스로 이동
	- Rename Method	- 메서드 이름이 목적을 드러내도록 이름 변경
	- Replace Temp Query	- 임시 변수 참조 시 메서드 호출로 교체

- SW 개발/운영시 개발자역량, 부족한 납기일정, 요구사항의 변동으로 코드스멜이 발생함.

2. 코드스멜의 정의와 특징

1) 코드스멜의 정의와 특징

개념	- 개발자의 역량 또는 부족한 일정으로 급히 코드를 구현 시 발생하며, 불필요한 변수, 주석, 스파게티코드 등으로 인해 코드가독성과 코드품질이 떨어지는 현상	
특징	개발자 역량 - SW 코딩 교육 부재 - 코드표준화의 부재	부족한 일정 - 잘못된 기간 산정 - 업무 프로세스 이해부족
	요구사항 변경 - 고객의 잦은 요청사항 - 정리되지 않은 암묵지 지식	코드수준 - 불필요한 주석 - 메서드의 집적 및 분할

- 코드스멜은 SW 동작가능 오류는 동작불가한 특징을 지니며, 높은 리팩토링 수준에 따라, 코드의 가독성, 퍼포먼스, 로직개선이 가능하므로 코드스멜 제거 및 기술부채방지.

2) 코드스멜과 리팩토링간의 상관관계

상관관계도	상관관계를 통한 해석
	<p>리팩토링 수준</p> <p>- 리팩토링 수준(반복)수가 높을 경우 코드품질 증가, 코드스멜 감소, 노력비용 감소 효과</p> <p>코드스멜</p> <p>- 코드스멜이 감소할수록, 개발/운영 인력의 코드 및 업무프로세스 이해 용이로 SDLC 증가 효과</p>

- 대표적으로 일상적으로 리팩토링이 가능한 중복코드, 긴 메서드, 많은 매개변수, 집중 변경이 존재

3. 코드스멜의 종류와 각각의 리팩토링 방법

1) 중복코드, 긴 메서드의 리팩토링 방법

코드스멜	리팩토링 기법	리팩토링 방법
중복코드 (Duplicated Code)	<ul style="list-style-type: none"> - Extract Method - Pull UP Method - Substitute Algorithm 	<ul style="list-style-type: none"> - 코드를 메서드로 분리 - 코드에 주석을 달아야만 의도를 이해 가능한 경우 - 메서드는 목적을 나타내는 직관적인 명칭을 사용 <div> <pre>void printOwing(double amount) { printBanner(); // print details System.out.println("name: " + _name); System.out.println("amount" + amount); }</pre> <pre>void printOwing(double amount) { printBanner(); printDetails(amount); }</pre> </div>
긴 메서드 (Long Method)	<ul style="list-style-type: none"> - Extract Method - Decompose Conditional 	<ul style="list-style-type: none"> - if, then, else 부분을 각각의 메서드로 분리 <div> <pre>if (date.before(SUMMER_START) date.after(SUMMER_END)) charge = quantity * winterRate + winterServiceCharge; else charge = quantity * summerRate;</pre> <pre>if (notSummer(date)) charge = winterCharge(); else charge = summerCharge();</pre> </div>

- 중복코드, 긴 메서드 유형의 코드스멜은 일상적으로 리팩토링을 수행하며, 개선 가능함.

2) 많은 매개변수, 집중 변경의 리팩토링 방법

코드스멜	리팩토링 기법	리팩토링 방법
많은 매개변수 (Long Parameter List)	<ul style="list-style-type: none"> - Introduce Parameter Object - Preserve Whole Object - Replace Parameter with Method 	<ul style="list-style-type: none"> - 매개변수를 제거하고, 직접 메소드 호출 - 매개변수전달 메서드를 전달받는 쪽도 호출 할 경우 - 호출한 메서드의 매개변수가 계산에 사용시 삭제불가 <div> <pre>public class Room { boolean withinPlan(HeatingPlan plan) { int low = daysTempRange().getLow(); int high = daysTempRange().getHigh(); return plan.withinRange(low, high); } }</pre> <pre>public class Room { boolean withinPlan(HeatingPlan plan) { return plan.withinRange(daysTempRange()); } }</pre> </div>
집중 변경 (Divergent Change)	<ul style="list-style-type: none"> - Extract Class 	<ul style="list-style-type: none"> - 클래스 일부를 분리해 새로운 클래스 생성 - 여러 클래스의 기능이 단일 클래스에 집중되는 경우 <div> <pre>public class Person { private String _name; private String _officeAreaCode; private String _officeNumber; public String getName() { return _name; } public String getTelephoneNumber() { return "(" + _officeAreaCode + ")" + _officeNumber; } }</pre> <pre>public class OfficeAreaCode { public String getOfficeAreaCode() { return _officeAreaCode; } public void setOfficeAreaCode(String arg) { this._officeAreaCode = arg; } public String getOfficeNumber() { return _officeNumber; } public void setOfficeNumber(String arg) { this._officeNumber = arg; } }</pre> </div>

- 프로젝트 초기부터 리팩토링 가이드 및 코드표준화를 고려하여, 코드작성시 SW 품질, 안전성 증대. "끝"

[추가자료]

구분	코드스멜	리팩토링 방법
클래스 측면 코드스멜	- 게으른 클래스 (Lazy Class)	- 자식 클래스와 부모 클래스 차이 없으면 합침
	- 추측성 일반화 (Speculative Generality)	- 추측만으로 생성한 클래스는 제거
	- 미들 맨 (Middle Man)	- 다른 클래스로 위임하는 역할만 담당하는 클래스 검토
객체 측면 코드스멜	- 메시지 체인 (Message Chain)	- 특정 객체를 얻기위한 다수 객체는 간소화
	- 거부된 유산 (Refused Bequest)	- 부모클래스의 I/F 거부 시 부모 객체 호출
	- 불완전 라이브러리 (Incomplete Library)	- 불완전 시 필요 부분 추가 구성
필드 측면 코드스멜	- 임시필드 (Temporary Field)	- 파라미터를 줄이기 위한 필드, 메서드는 클래스화
	- 부적절한 친밀 (Inappropriate Intimacy)	- 불필요 데이터를 가지고 있는 경우
	- 주석(Comment)	- 코드 내부 주석은 메서드로 추출해서 설명

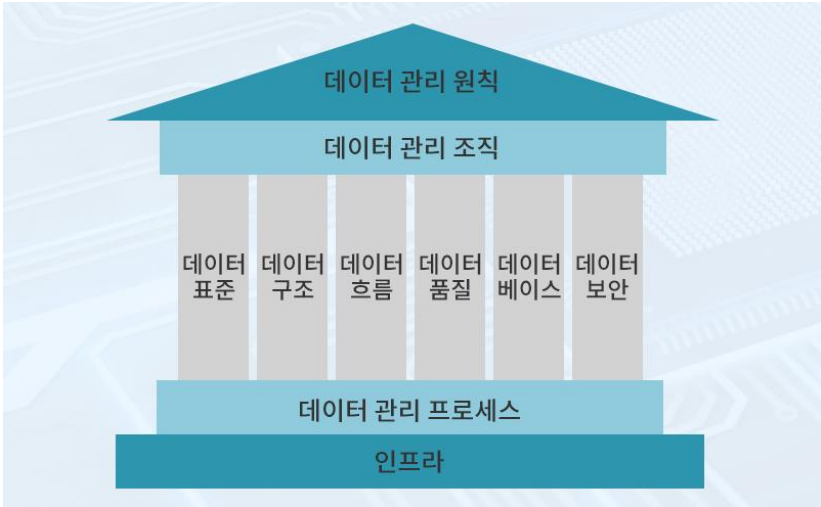
기출풀이 의견

2. 리팩토링과 코드스멜의 연관관계를 이해후 세부 리팩토링 기법 및 개선방안에 대해 고민하시면 좋겠습니다.

문 제	2. A기업에서는 비즈니스 수행과정에서 수집된 많은 양의 빅데이터(Bigdata)를 통합 관리하고자 한다. 데이터 관리에 대한 아래의 사항을 설명하시오. 가. 데이터 거버넌스(Data Governance)의 개념 및 주요 기능 나. 마스터 데이터(Master Data)의 개념과 필요성 다. 마스터 데이터 관리(MDM)의 구성요소와 구축 시 고려사항		
출 제 영 역	DB	난 이 도	★★☆☆☆
출 제 배 경	- 기업의 데이터의 중요성 증가		
출 제 빈 도	- 124 회 관리 2 교시, 모의고사/합숙 다수 출제		
참 고 자 료	- 데이터 거버넌스 – ADSP - 공공과 민간이 하나되는 데이터 거버넌스 전략 – 코스콤		
Key word	- 데이터거버넌스(정책, 절차, 역할, 책임), 데이터 품질관리, 메타데이터 관리, 데이터 주기관리, 데이터 보안 및 프라이버시		
풀 이	이정현(125 회 정보관리기술사)		

1. 데이터 거버넌스의 개념 및 주요 기능

1) 데이터 거버넌스의 개념

개념	<ul style="list-style-type: none"> - 광의적 측면 : 데이터 정책, 절차, 역할, 책임 등을 조직하고 실행하는 활동으로 의사결정과 규정을 적용하여 <u>효율적인 정보자산을 관리</u>하는 데이터 관리 체계 - 협소적 측면 : 데이터 품질 보장, 프라이버시 보호, 데이터 수명관리, 전담조직 규정 정립, 데이터 소유권과 관리적 명확화 등을 통해 데이터가 필요한 사람에게 적시적으로 제공 가능토록 체계를 확립하는 활동 - 빅데이터 측면 : 전사차원 빅데이터 분석을 기업문화로 정착시키고 데이터 분석업무를 지속적으로 고도화하기 위해 <u>데이터, 프로세스, 시스템, 역량, 조직</u> 등으로 구성된 빅데이터 관리 체계
데이터 거버넌스	

- 데이터의 가용성, 유용성, 통합성, 보안성 관리를 위한 데이터 거버넌스 기능이 존재.

2) 데이터 거버넌스의 주요 기능



- 데이터 품질, 메타데이터, 데이터 인벤토리, 데이터 수명주기, 데이터 액세스 및 권한 부여, 데이터 통합 등의 통제 기능들을 포함한 데이터 관리
- 데이터 거버넌스에서 전사관점의 기준정보 관리를 통해 정확성, 일관성을 보장하기 위해 마스터 데이터 사용

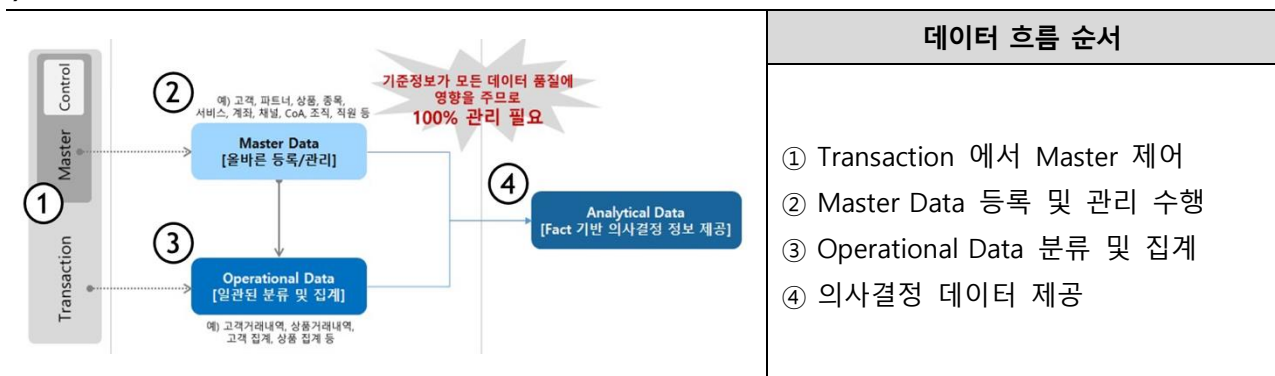
2. 마스터데이터의 개념과 필요성

1) 마스터 데이터의 개념과 필요성

개념	- 업무, 프로세스에서 공통참조 및 활용되는 전사관점의 기준정보와 <u>기업의 일관된 View</u> 를 제공하는 기준 데이터
필요성	<ul style="list-style-type: none"> - 조직 전반 업무의 정확성과 일관성을 보장하는 기준 데이터 - 변경 가능성이 적고, 조직 전반에 공유되므로 최신의 Single View 유지 필요. - 메타데이터 수집 및 수립 후 마스터 데이터 관리 수행 - 빅데이터의 메타데이터 활용 목적으로 사용

- 마스터 데이터(기준정보)는 전사시스템 및 데이터 품질에 영향을 주므로 데이터 흐름 파악 후 관리 수행

2) 마스터 데이터의 데이터 흐름



- 마스터데이터 흐름, 정합성, 표준화를 위해 마스터 데이터 관리를 활용.

3. 마스터 데이터 관리의 구성요소와 구축 시 고려사항

1) 마스터 데이터 관리의 구성요소

구분	구성요소	설명
거버넌스	- 거버넌스 정책	- 데이터 품질관리, 표준화, 품질진단, 조직구성
	- 조직체계	- 전사데이터 관리자(CIO), 업무데이터 관리자(DA), DBA
	- 프로세스	- 데이터 계획, 통제, 보증, 개선 반복 수행
IT 인프라	- 연계 I/F	- 전사 기준정보 및 운영업무 데이터 동기화 및 표준화
	- EAI / ETL	- 데이터 추출, 변형, 정제, 전처리 I/F, 로그, 크롤링
	- Data Storage	- 마스터 데이터 모델링, 데이터 저장소, OLAP 활용
운영 표준	- 표준 체계	- ISO8000, 데이터 분류, 속성, 식별, 표기 표준 체계
	- 품질 체계	- DQM, 품질기준, 품질관리 프로세스, 품질 성숙도
	- 데이터 통합	- MDM 을 위한 조직내 기준정보 지속 통합 및 관리

- 마스터 데이터 구축 시 대상, 기준정보분류, 선정기준의 체계적 기준수립이 선행되어야함.

2) 마스터 데이터 구축 시 고려사항

마스터데이터 대상	마스터데이터(기준정보)		선정 기준 항목	판단 내용
	① 기준정보 대상	② 기준정보 분류	③	
		마스터 기준정보 • 전사 업무에 동일한 기준으로 사용되는 핵심 업무 데이터 • 예 : 고객, 상품, 서비스, 종속 등 컨트롤 기준정보 • 데이터의 입력/집계/분석 시 편의를 위해 설계된 코드 • 예 : 고객구분코드, 자산관리유형코드 등 운영 기준정보 • 업무프로세스 실행결과 또는 산출식에 의해 생성 변경, 폐기되는 데이터로 참조되거나 입력값으로 활용 • 예 : 수수료 징수율, 약정이자율 등	관리 범위 대상 여부	• 마스터 관리 범위 내에 포함되는 정보인지?
			활용도	• 전사적으로 활용하는 정보인지?
			중요도	• 업무적, 전략적 중요 정보인지?
			정보 획득 가능성	• 정보 획득 절차, 시점, 주체가 명확하고 고객 또는 외부에서 획득 가능한 정보인지?
			데이터 정확도 보장성	• 업무에서 활용 가치가 있는 정보로 제공가능한지?
			가공 여부	• 트랜잭션 결과를 가공한 속성인지?
			지속 관리 여부	• 장기적으로 유지관리 가능한 속성인지?
고려사항	① 기준정보 대상의 전사정보와 기준정보의 중복제거 및 일관화의 선행화 ② 기준정보 분류를 마스터, 컨트롤, 운영 정보로 체계화하여 분류 수행 ③ 기준정보 분류대상의 대상여부, 활용도, 중요도, 정확도를 고려하여 구축			

- 마스터 데이터의 구축/관리 체계화와 지속 축적을 통해 데이터 거버넌스의 발전 및 인사이트 도출. "끝"

기출풀이 의견

3. 목차구성이 매우 어려운 문제였습니다. 최근 공공/민간 데이터 거버넌스를 통해 새로운 가치 창출 및 인사이트를 도출을 위한 정책을 펴고 있어, 앞으로도 충분히 출제가능성이 있는 토픽으로 보입니다.

문 제	4. 데이터베이스 옵티마이저(Optimizer)에 대한 아래 사항을 설명하시오. 가. 옵티마이저의 개념 나. RBO(Rule Based Optimizer)와 CBO(Cost Based Optimizer) 비교 다. 옵티마이저의 적용 시 고려사항		
출 제 영 역	DB	난 이 도	★★☆☆☆
출 제 배 경	- 데이터의 중요성, 복잡성의 증가로 출제		
출 제 빈 도	- 110 회 응용 1 교시, 92 회 응용 2 교시, 모의고사/합숙에서 다수 출제		
참 고 자 료	- 데이터베이스 옵티마이저(Optimizer)에 대하여 - 코딩팩토리		
K e y w o r d	- DBMS 핵심엔진, 실행계획최적화, RBO(규칙, 연산자 분류, 인덱스), CBO(비용, 통계정보)		
풀 이	이정현(125 회 정보관리기술사)		

1. 옵티마이저의 개념

개념	개념도
<ul style="list-style-type: none"> - SQL Query의 수행 효율성을 위해 RBO, CBO 사용하는 DBMS 핵심 엔진 또는 실행계획 최적화 기법 - SQL을 수행할 최적의 처리 경로를 생성해주는 DBMS의 핵심 엔진 	<pre> graph TD User[사용자] -- SQL --> Parser[파서] Parser --> Optimizer[옵티마이저] subgraph Optimizer Rule[규칙기반 옵티마이저] Cost[비용기반 옵티마이저] end Cost <--> 통계 Dict[딕셔너리] Optimizer -- 실행계획 --> Engine[SQL 실행 엔진] Engine -- 결과 --> User </pre>

- 옵티마이저는 RBO와 CBO로 구분되며, 실행계획, 통계정보 유무 등 차이가 존재

2. RBO(Rule Based Optimizer)와 CBO(Cost Based Optimizer) 비교

1) RBO와 CBO의 개념적 비교

비교항목	RBO	CBO
개념	- 실행 속도가 빠른 순으로 규칙을 먼저 세워두고 우선순위가 높은 실행계획을 수행하는 최적화 기법(오라클 8 이하)	- 실행 계획을 세운 뒤(최대 2천개까지) 비용이 최소한으로 나온 실행 계획을 수행하는 최적화 기법(오라클 10 이상)
관점	- Single Row, Hash, Full Table Scan 등 지정된 규칙으로 수행	- 테이블, 인덱스, 컬럼 등 처리방법의 비용산정 후 최소비용을 선택하여 수행
등장	- 오라클 8 이하	- 오라클 10 이상(11부터 CBO만 사용)
통계정보 유무	- 미존재	- 존재

- RBO의 성능문제를 개선하여 등장한 것이 CBO이며, 통계(Dictionary)통해 비용산정 후 쿼리 수행

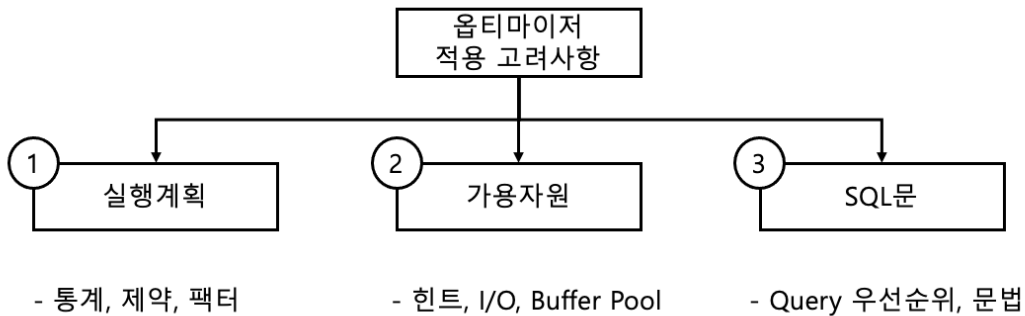
2) RBO 와 CBO 의 상세 비교

비교항목	RBO	CBO
실행계획	- 미리 정해진 규칙으로 실행계획 결정 - 연산자, 접근 경로, Syntax 규칙으로 지정	- SQL 의 소요비용 예측후 실행계획 결정 - I/O 횟수, CPU, Mem, N/W 사용량 고려
소요비용	- 비용 고려치 않음(소요비용 높음)	- H/W 관점 비용 고려(상대적 낮음)
SQL 실행 예측	- SQL 실행절차 미리 예측 가능(정적)	- 통계정보의 변동으로 예측 어려움(동적)
오차가능성	- 통계정보 없어, 오차발생 증가	- 통계정보 현행화로 오차발생 감소
성능	- 힌트, 해시조인 사용불가	- RBO 의 문제개선 및 최소 성능 보장

- Select *From emp where empno = 7900; 실행시 RBO I/O 는 9 회, CBO I/O 는 5 회 발생하므로 CBO 우수

3. 옵티마이저 적용 시 고려사항

1) 옵티마이저 고려사항 유형



- 옵티마이저 적용시 실행계획, 가용자원, SQL 문의 고려가 필요.

2) 옵티마이저 적용 시 고려사항

구분	고려사항	설명
1. 실행계획	- 통계정보	- CBO 의 모든 판단 기준은 통계정보에 근거
	- 제약설정	- 개체, 참조, 도메인, 사용자 정의 무결성 규칙 고려
	- 팩터	- 인덱스, 클러스터링, 파티셔닝 등 팩터 고려
2. 가용자원	- 힌트	- 잘못된 참조(인덱스, 테이블) 문법 주의하여 힌트작성
	- I/O 최소화	- 실제 필요한 데이터만 조회하여 I/O 최소화
	- Buffer Pool 튜닝	- 반복 작업 및 오라클 Keep Buffer Cache 활용
3. SQL 문	- Query 질의	- 뷰, 조건절, 서브쿼리, 조인 기법을 유의하여 질의수행
	- SQL 과 연산자 형태	- 연산자 우선순위 고려(=, IN, LIKE, BETWEEN, >, <)
	- Parser	- SQL Syntax 문법 오류, Semantic 맥락을 고려

- 옵티마이저 적용 시 DB 설계, DBMS 선택, 튜닝등 다양한 관점에서 접근 필요. "끝"



기출풀이 의견

4. 옵티마이저, Hint 등 DB도메인이 자주 출제되고 있습니다. 가장 좋은 흐름은 옵티마이저 > 옵티마이저 원리 > Parser(Soft/Hard) > RBO, CBO > 실무경험 순이며, 나만의 인사이트를 가지면 좋겠습니다.

문 제	5. 소프트웨어 정의 네트워크(SDN)에 대한 아래의 사항을 설명하시오. 가. SDN 제어평면의 개요 및 구조의 특징 나. 오픈플로우(OpenFlow) 프로토콜		
출 제 영 역	NW	난 이 도	★★★★☆☆
출 제 배 경	- 벤더 종속 탈피, 네트워크 리소스 최적화, Biz 변화 대응		
출 제 빈 도	119 회 응용 4 교시, 114 회 응용 2 교시, 모의고사 합숙 다수 출제		
참 고 자 료	- SDN/OpenFlow 기술 동향 및 전망 - KNOM		
Key word	- SDN(제어평면, 데이터평면, 벤더 탈피, 관리효율화), Openflow(Controller, Switch, Secure Channel)		
풀 이	이정현(125 회 정보관리기술사)		

1. SDN 제어평면의 개요 및 구조의 특징

1) SDN 제어평면의 개요

개요도			
SDN 제어평면 개념	- 네트워크 제어기능(Control Plane)과 데이터 전송기능(Data Plane)을 분리, Open API(OpenFlow) 이용한 SW기반 NW 구성 및 제어 기술		
Openflow 개념	- 네트워크 스위치(혹은 라우터)의 패킷포워딩 기능과 제어기능을 분리하고 이들 간 통신을 가능케하는 프로토콜 기술		

- SDN 는 제어평면, 데이터평면, 관리평면으로 구성되고, 구조는 Application, Control, Infrastructure Layer 로 구성

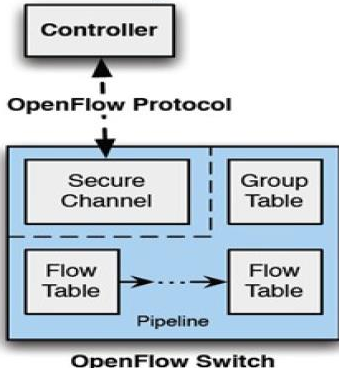
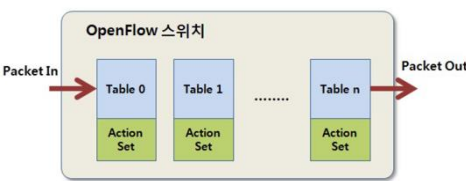
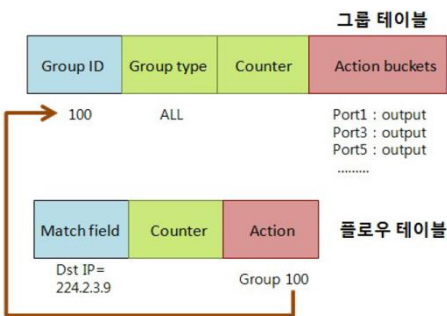
2) SDN 제어평면의 구조의 특징

구분	특징	설명
계층 측면	- Application Layer	- 비즈니스, SDN 응용 등 사용자 접근 가능한 계층
	- Network Control Layer	- 주요 핵심요소인 SDN 컨트롤러가 위치하는 계층
	- Infrastructure Layer	- WAF, ESM, 방화벽, L3 등 NW 장치를 관리하는 계층
관리 측면	- SDN 표준화	- 포워딩시 개방형 API와 I/F를 통해 제어
	- SDN 글로벌 관리 추상화	- 네트워크 기능 SDN컨트롤러에 집중화
	- 통합 및 N/W 관리	- 데이터 센터내 VM, 스토리지 등 자원 관리 일원화

- SDN 구성을 위해 OpenFlow 를 사용하며, 특히 OpenFlow 프로토콜은 구성요소와 동작과정이 매우 중요함

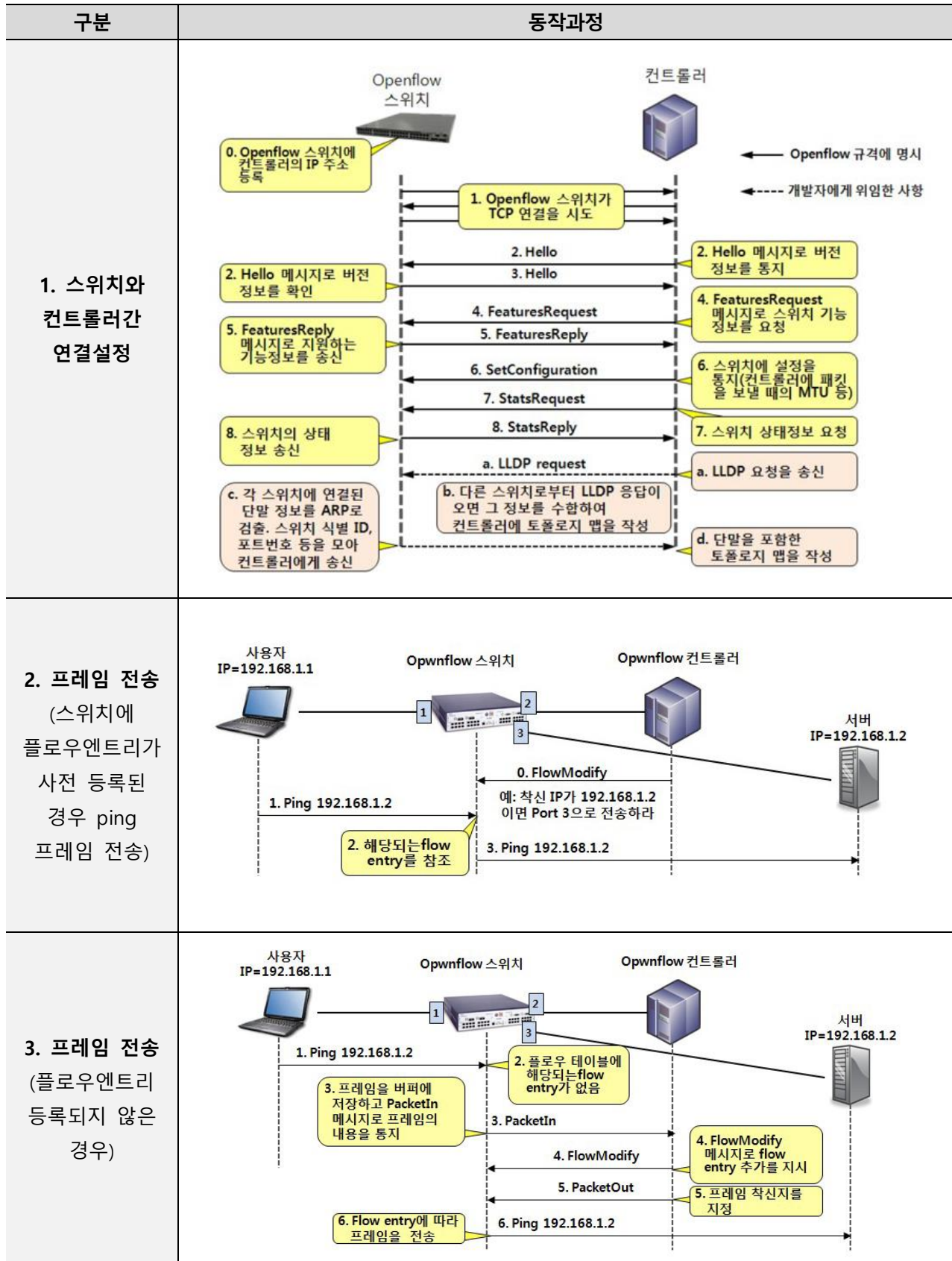
2. 오픈플로우 프로토콜

1) 오픈플로우 프로토콜의 구성요소

구성요소	구성도	설명 / 설정												
1. 컨트롤러, 스위치		<ul style="list-style-type: none">- 컨트롤러 : 스위치에 명령 전달 수행- 스위치 : 패킷 전송, 수정, 폐기 수행												
		<ul style="list-style-type: none">① L2 스위치에 Openflow 펌웨어 추가② Openflow 스위치 구성③ 컨트롤러는 SW로 구현												
2. 플로우 테이블	<p>플로우 테이블의 내용</p> <table border="1"><thead><tr><th>Flow entry</th><th>조건 (match fields)</th><th>통계 (counters)</th><th>처리(action)</th></tr></thead><tbody><tr><td>1</td><td>이 패킷의...</td><td>현재상황은...</td><td>이렇게 처리 하라</td></tr><tr><td>2</td><td>...</td><td>...</td><td>...</td></tr></tbody></table>	Flow entry	조건 (match fields)	통계 (counters)	처리(action)	1	이 패킷의...	현재상황은...	이렇게 처리 하라	2	<ul style="list-style-type: none">- 플로우엔트리 : 조건, 처리, 통계로 구성- 플로우테이블 : 플로우 엔트리의 집합
Flow entry	조건 (match fields)	통계 (counters)	처리(action)											
1	이 패킷의...	현재상황은...	이렇게 처리 하라											
2											
3. 파이프라인 처리		<ul style="list-style-type: none">- 파이프라인처리 : 플로우 테이블에 번호가 0부터 순차대로 증가하고, 플로우 테이블 존재시 통계+1, 처리 명령 실행												
		<ul style="list-style-type: none">① 플로우 테이블에 번호 부여(0~n)② 입력패킷과 플로우테이블 비교③ 매치될 경우 통계값+1 후 명령 실행④ 처리(action) 반복 수행												
4. 그룹테이블		<ul style="list-style-type: none">- 그룹테이블 : 특정 플로우의 Action Bucket을 실행 할 때 사용하는 테이블												
		<ul style="list-style-type: none">① Broadcast, Multicast 유형 선택② 플로우테이블에 IP 목적지 지정③ 파이프라인 처리												

- Openflow 스위치를 Openflow 네트워크에 최초설치시 IP 주소 지정 필요. 특히 Openflow 동작과정으로 기술

2) 오픈플로우 프로토콜 동작과정



- SDN 컨트롤러 통해 중앙집중관리 가능하나, 확장성, 구조적, 구축 문제가 존재.

3. SDN 구축시 문제 및 해결방안

문제점	해결방안
확장성 문제 (하나의 스위치에서 수십만개의 플로우 관리 어려움)	- NFV 를 활용한 관리영역 제어 - 리눅스 & IP N/W 활용
구조적 문제 (스위치 고장시 링크우회 문제)	- WAN 기술인 SDH/SONET 통해 빠른 Failover 유도 - Openflow 컨트롤러 데이터센터내에서 관리
네트워크 구축 문제 (SDN 구축역량 부족)	- TRILL(TRansparent Interconnection of Lots of Links), SPB(Shortest Path Bridging)을 활용 - Flat 구조 활용하여, N/W 계위 최소화

- SDN, OpenFlow, RouteFlow, LegacyFlow, NFV 은 융합되어, 집중제어의 장점과 분산제어 확장성을 모두 갖춘 자율제어 네트워킹 기술로 발전할 것으로 기대.



기출풀이 의견

5. SDN, SDI, SDC 등 관련 토픽을 연계해서 보시면 더욱 좋을 것 같습니다. 특히, 구성요소/동작과정은 정확히 적는다면 차별화 되실 것으로 판단됩니다.

문 제	6. SOAR(Security Orchestration, Automation and Response)의 개념 및 등장배경, 구성요소, 주요기능, 기대효과, 도입 시 고려사항에 대하여 설명 하시오.		
출 제 영 역	보안	난 이 도	★★☆☆☆
출 제 배 경	- 기존 보안관제체계의 정보자산의 안전 보장이 어려움		
출 제 빈 도	127 회 관리 3 교시 첫 출제, 모의고사/합숙 다수 출제		
참 고 자 료	- 보안사고 SOAR 로 대응하자 - LG CNS - 보안 관제의 디지털 탈바꿈, SOAR 의 개념과 주요 기능 - 2e 컨설팅		
K e y w o r d	- SOC, SOA, SIRP, TIP, 오케스트레이션, 자동화, 사고대응 및 협업, 리포팅, 보안관제 3 세대		
풀 이	이정현(125 회 정보관리기술사)		

1. 차세대 보안관제, SOAR 의 개념 및 등장배경

1) SOAR 의 개념

개념	개념도
- 보안 오케스트레이션 및 자동화(SOA) , 보안사고 대응(SIRP), 위협 인텔리전스 플랫폼(TIP)을 통합하여 효과적 보안관제가 가능한 3 세대 보안 솔루션	

- 기존 보안관제체계(ESM, SIEM) 방식의 Known/UnKnown Attack 의 제한적 대응 및 한계해결 위해 SOAR 등장

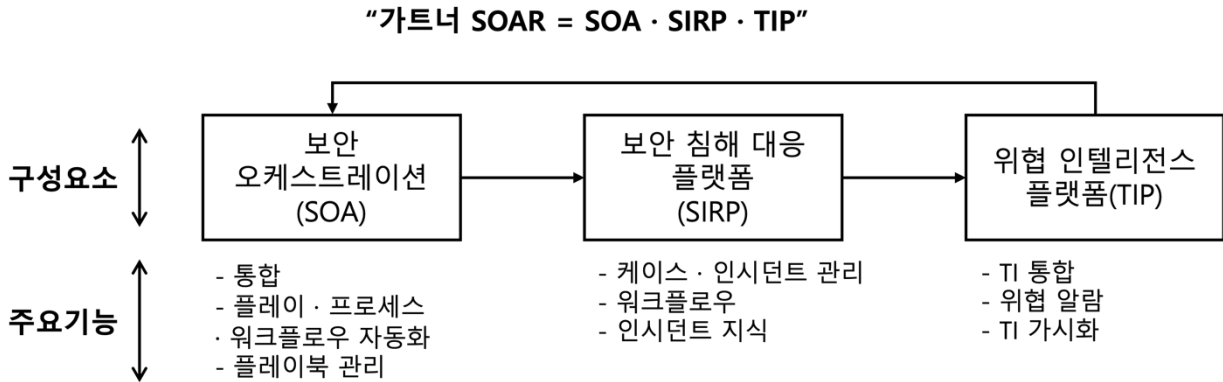
2) SOAR 의 등장배경

① 기존 보안관제 한계 해결 - 정보수집, 분석, 대응조치, 보고 절차를 자동화 - 위협대응 의사결정 지원	③ 정보자산 보안안전 및 효율성 증가 - 정보자산 패킷 흐름 파악 용이 - 보안위협 발생시 조직원의 집중 효율화
② 사이버 공격 복잡화 - 지능형 표적공격, 클라우드 해킹공격 - 스피어피싱 + 랜섬웨어	④ 의사결정 데이터 제공 - 전사 보안정책 통합 및 보안의사결정 도움 - 정보기기의 패킷 흐름 파악 용이

- 디지털 전환 가속으로 신기술을 이용해, 지능적이고 고도화된 AI 기반 Poisoning, Evasion Attack 등이 보고됨.
 - SOAR 는 3 가지를 통합한 서비스 형태로 보안인력의 보안대응 집중 및 효율화에 큰 도움을 주는 것으로 파악

2. SOAR의 구성요소 및 주요기능

1) SOAR의 구성요소 및 주요기능



- SOAR는 SOA, SIRP, TIP로 구성되며, 주요기능으로 플레이북, 워크플로우, 오케스트레이션, TI를 제공

2) SOAR의 구성요소 및 주요기능 설명

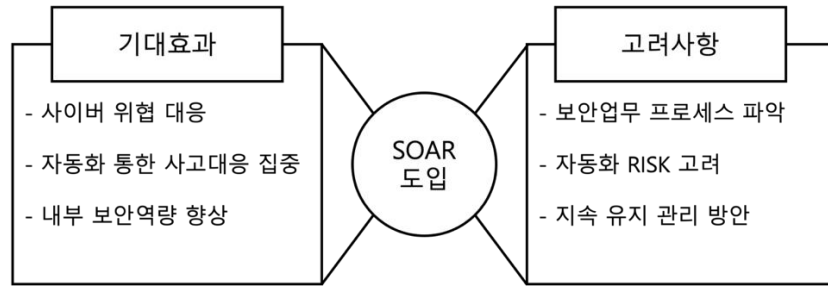
구분	고려사항	설명
구성요소	보안 오케스트레이션	- 보안 통합 관리, 플레이북 관리, 워크플로우 자동화 - WAF, App 보안, 클라우드 보안, EDR, NDR 연계
	보안침해대응 플랫폼	- 비인가 시스템 접근 통제(NAC), 네트워크 패킷 차단 - 프라이버시 보호, 서비스 지속성 보장, 취약점 분석
	위협 인텔리전스 플랫폼	- IoC(Indicator Of Compromise) 분류, 스코어링 제공 - 화이트리스트 관리, TI(Threat Intelligence) 통합 가시화
주요기능	Playbook	- 사이버 공격 대응 및 실시간 보안 사고 대응 - 보안경고, 인시던트 최소화, 자동화 기반 이벤트 관리
	Workflows	- 워크플로우 시각화(다이어그램, 차트, 흐름도) - 보안 규정 준수 문제 감지 및 해결(비용 절감)
	Visualization	- 이벤트 간 IOC 시각화 및 조직의 침해지표 판단 - 보안 현황 가시화, 보안 시각화, 신속 대응 유도
	Task	- 스크립트 기반 자동화 대응 - Unification, Orchestration, Automation 작업

- SOAR는 보안인력의 업무최소화를 유도하므로 비용절감 및 보안역량 최대화를 보장

- 특히 SOAR 도입할 경우 업무, Risk, 관리성을 고려할 시 기대효과 상승 시너지 발생

3. SOAR의 기대효과 및 도입 시 고려사항

1) SOAR의 기대효과 및 도입 고려사항



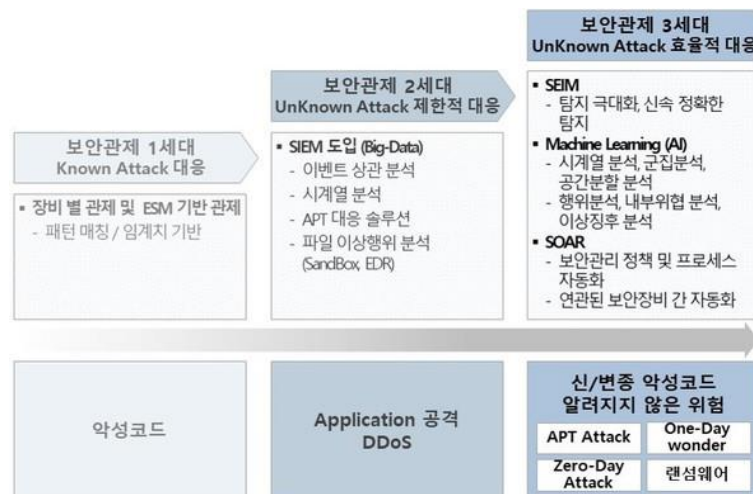
- SOAR 도입시 보안 환경분석, 현황분석, 목표모델 검토후 SOAR 도입이 필요.

2) SOAR 기대효과 및 고려사항 설명

구분	주요 측면	설명
기대효과	- 전사 측면	- 보안 기술 통합 제공으로 사이버 위협 대응 - 보안 기술 투자 대비 ROI 향상
	- 실무자 측면	- 플레이북 기반 사고 유형별 최적화된 프로세스 생성 - 자동화를 통한 시간 절약 및 사고 대응에 집중
	- 관리자 측면	- 사고 대응 프로세스 최적화로 내부 역량 강화 - 보안 역량 상향 평준화 및 업무성과 가시화
고려사항	- 보안업무 프로세스	- 기존 보안업무 프로세스의 자동화를 위한 환경분석 - SIEM, EDR 연계 시스템과의 영향 고려
	- 자동화 Risk	- 예방, 감지, 대응, 분석, 보고 프로세스의 연계 고려 - As-Is, To-Be 분석 자동화 및 프로세스마이닝 고려
	- 지속 유지관리	- 플레이북 디자이너, SORA 개발자, 보안툴 유지관리 - 오케스트레이션 프로세스 관리 전담 Task 고려

- 보안업무, Risk, 지속성 고려시 SOAR 도입 기대효과는 증가함. 추가로 보안관제서비스의 트렌드 고려 필요.

4. 보안관제서비스의 발전방향



- 기존 보안관제, SIEM, SOAR 등을 연계하여, APT, Zero-day, 랜섬웨어 위협을 최소화 가능 "끝"



기출풀이 의견

6. 기존 보안관제 문제 해결위해 등장한 SOAR는 지능적으로 진화하는 보안공격에 효율적인 서비스 및 플랫폼으로 볼 수 있습니다. 앞으로 충분히 출제 가능한 토픽이므로 보안 키워드와 함께 준비하면 좋겠습니다.