119 회 기출풀이

정보관리기술사

- KPC 기술사회 -

SMART KPC
한국생산성본부

교육 문의 및 상담 : 한 승 연

- Tel : 02) 724-1831/1223 - Fax : 02) 724-1875
- Email : syhan@kpc.or.kr
- Web Site : www.kpc.or.kr
cafe.naver.com/81th

kpc 기술사회

120 회 합격대비 심화반 신청 안내

[토요일 명품심화반]

- O 단합반(SPP 반) (안경환 PE @ KPC) : 정규심화 9. 28. 개강
- O FB(Future Builders) (강희석 PE @ KPC) : 정규심화 9. 28. 개강
- O 열 정 반 (박상욱 PE @ KPC) : 정규심화 9. 28. 개강
- O 정 주 행 (서정훈 PE @ KPC) : 정규심화 9. 28. 개강
- O ITPE Makers (박제일 PE @ KPC) : 정규심화 9. 28. 개강
- O MP 필통반 (구환회 PE @ KPC) : 정규심화 9. 28. 개강
- O 공 감 반 (공수재 PE @ KPC) : 조기심화 8. 31. 개강

[일요일 명품심화반]

- O T.O.P 반 (유술사 PE @ KPC) : 조기심화 8. 25. 개강
- O NS 반 (강정배 PE&박주형 PE @ 강남아지트): 조기심화 9. 1. 개강

[평일 명품심화반]

- O 강남평일야간반 (강정배 PE&전일 PE&박찬렬 PE @ 강남아지트/화,금):
  조기심화 9. 3. 개강

~~ KPC 홈페이지에서 바로 신청 가능합니다. ~~

국가기술자격 기술사 시험문제

기술사 제 119 회 제 1 교시 (시험시간: 100분)

분야	정보통신	종목	정보관리기술사	수험번호		성명	

청렴이 세상 함께해요~ 청렴실천 같이해요!! 청정한국!! 한국산업인력공단

※ 다음 문제 중 10문제를 선택하여 설명하시오. (각10점)

1. COCOMO Ⅱ(Constructive Cost Model Ⅱ)의 Post-Architecture Model
2. 제약이론 TOC(Theory of Constraints)
3. 가상화기술의 도커(DOCKER)
4. DID(Decentralized Identity)
5. OTT(Over-The-Top) 서비스의 제로 레이팅(zero-rating) 문제점
6. UX기법인 Design Thinking
7. Data Lake
8. 데이터베이스 샤드(database shard)
9. 기계학습(Machine learning)의 Gradient descent algorithm
10. Lehman의 SW 변화원리
11. 머클트리(Merkle Tree)
12. 리쇼링(Reshoring)
13. 데이터베이스의 Isolation Level

1 - 1

1	Post-Architecture Model
문제	COCOMO II (Constructive Cost model II)의 Post-Architecture Model
도메인	소프트웨어공학
정의	가장 세부적인 COCOMO II 모델로, 구조 설계 이후 단계에 시스템에 대한 자세한 이해를 바탕으로 COCOMO 에서 제안된 LOC 에 의하여 소요되는 노력을 추정하는 노력 추정방법
키워드	Multiplicative cost, Scaling cost driver, Cost Driver (Very Low ~ Extra High)
출제의도분석	전통적인 노력 추정 방식인 COCOMO 의 심화형 문제, 기 출제 영역인 COCOMO 이외의 COCOMO2 의 상세 기법 질의
답안작성 전략	COCOMO2 기반 답안지 작성, 단어 유추를 통한 3 단계에서의 추정
참고문헌	COCOMO II Model Definition Manual (University of Southern California) 소프트웨어 공학 5 차 개정판 (최은만)
풀이 기술사님	문광석 PE(제 117 회 정보관리기술사 /itpe.moon@gmail.com)

1. COCOMOII 의 구조 설계 이후 단계, COCOMO II 의 Post-Architecture Model 의 개념

- 가장 세부적인 COCOMO II 모델로, 구조 설계 이후 단계에 시스템에 대한 자세한 이해를 바탕으로 COCOMO 에서 제안된 LOC 에 의하여 소요되는 노력을 추정하는 노력 추정 방법

2. Post-Architecture Model 의 기본 산식 및 상세구성

가. Post-Architecture Model 의 기본 산식

계산식

$$PM = \prod_{i=1}^{17} (EM_i) \cdot A \cdot \left[\left(1 + \frac{REVL}{100}\right) \cdot Size\right]^{\left(0.91 + 0.01 \sum_{j=1}^{5} SF_j\right)} + \left(\frac{ASLOC \cdot \left(\frac{AT}{100}\right)}{ATPROD}\right)$$

where

$$Size = KNSLOC + \left[KASLOC \cdot \left(\frac{100 - AT}{100}\right) \cdot \frac{(AA + SU + 0.4 \cdot DM + 0.3 \cdot CM + 0.3 \cdot IM)}{100}\right]$$

$$B = 0.91 + 0.01 \sum_{j=1}^{5} SF$$

ASLOC : 새로 생성할 코드, AT : 자동 생성되는 부분, ATPROD : 통합시 필요한 생산성, KNSLOC : 새 소스코드의 라인으로 표현되는 컴포넌트의 사이즈, KASLOC : 적응형 LOC 로 표현되는 적응형 컴포넌트의 사이즈

약식

$$PM = A \times [Size]^{B} \times \prod_{i=1}^{17} EM_i + PM_M$$

PM : 추정되는 노력의 공수, A : 상수, PM_M : (ASLOC*(AT/100)/ATPROD)
EM : Multipliers (RELY, DATA, CPLX, RUSE, DOCU, TIME, STOR, PVOL, ACAP, PCAP)

- 1 단계 **Application Composition(응용 합성 모델)**을 통해 프로토 타이핑이후, 2 단계 **Early Designed Model(초기 설계모델)**을 통해 예측하고, 최종 3 단계 **Post-Architecture Model** 추정

나. Post-Architecture Model 의 상세구성

상세구성	설명	특징
A	- 상수 (일정한 계산을 위한 준비된 값)	사전에 미리 정의
SIZE	- 새로운 라인수+ 재사용 모델에서 수정할 라인수 + 요구사항 변경에 따른 라인수	LoC(라인수)기반의 Size
B	- 이전 경험(1:익숙 ~ 5:미경험),개발 유연성 (1:유연함 ~ 5:고정), Risk 관리(1:관리 ~ 5:미관리),팀워크(1:좋음 ~ 5:안좋음),프로세스 성숙도(1:성숙 ~ 5:미성숙) Ex) B= (요소의 총합) / 100 + 1.01 = 1.14	의존값의 총합에 대한 계산식을 통해 표준화
EM	- M(1)+M(2)+...+M(17)의 Multipliers Cost Drivers 의 총합	17 개의 Drivers
LoC Counting Rules	- COTS(상용소프트웨어), GFS(정부 공급 소프트웨어), 언어 지원 라이브러리 및 운영체제, 상업용 라이브러리, 소스코드 생성기로 생성된 코드 제외	정의시 미포함 영역 표현

- COCOMO II 모델의 **세 가지 단계의 순차적인 적용**을 통해 전체 노력의 상세 추정이 가능

3. COCOMO II 모델의 세 가지 단계 비교

비교 대상	Application Composition	Early Designed Model	Post-Architecture Model
크기	응용 포인트	기능 포인트(FP)와 언어 종류	FP 와 언어 LOC
재사용	모델에 포함	LOC 를 다른 변수의 함수로 사용	LOC 를 다른 변수의 함수로 사용
요구변경	모델에 포함	변경 비율이 비용승수로 반영됨	변경 비율이 비용승수로 반영됨
유지 보수	응용 포인트 연평균 변경 비율(ACT)	ACT, 이해력, 친밀성의 함수	ACT, 이해력, 친밀성의 함수
노력 예측 C 값	1.0	선행작업, 적응도, 초기 설계, 위험제거, 팀 결집력, SEI 프로세스 성숙도 따라 0.91 ~ 1.23	선행작업, 적응도, 초기 설계, 위험제거, 팀 결집력, SEI 프로세스 성숙도 따라 0.91 ~ 1.23
제품 비용 승수	없음	복잡도, 재사용 요구도	신뢰도, DB 규모, 문서화 정도, 재사용요구, 복잡
플랫폼 비용 승수	없음	플랫폼 난이도	실행시간제약, 기억공간제약, 가상기계
인력 승수	없음	개인 능력과 경험	분석, 응용 경험등
프로젝트 비용 승수	없음	개발 기간, 개발 환경에 대한 요구	도구 사용, 개발 기간, 여러 사이트 개발 요구

- 점차 단계가 진행될수록 구조 설계 이후에 상세화 되어 LOC 수준의 Detail 한 노력 추정 가능

"끝"

[참고자료] 17 개 Multipliers Cost Drivers (EM)

분류	구성	등급
Product Factors	Required Software Reliability (RELY)	-Very Low, Low, Nominal, High, Very High, Extra High 까지 6 개의 분류 등급으로 구성
	Data Base Size (DATA)	
	Product Complexity (CPLX)	
	Required Reusability (RUSE)	
	Documentation match to life-cycle needs (DOCU)	
Platform Factors	Execution Time Constraint (TIME)	
	Main Storage Constraint (STOR)	
	Platform Volatility (PVOL)	
Personnel Factors	Analyst Capability (ACAP)	
	Programmer Capability (PCAP)	
	Applications Experience (AEXP)	
	Platform Experience (PEXP)	
	Language and Tool Experience (LTEX)	
	Personnel Continuity (PCON)	
Project Factors	Use of Software Tools (TOOL)	
	Multisite Development (SITE)	
	Required Development Schedule (SCED)	

2	TOC
문제	제약이론 TOC(Theory of Constraints)
도메인	경영
정의	앨릿 골드렛 박사가 창시한 시스템의 목적(Goal) 달성을 저해하는 제약 조건을 찾고, 통제하여 전체를 최적화를 달성하는 상식적인 의사결정 지원 경영과학 이론
키워드	집중개선프로세스 5 단계, 사고 프로세스, Throughput Account, DBR, Critical Chain, 사건의 종속성, 통계적 변동, 전체 최적화, 제약, 병목
출제의도분석	경영분야에 재조명 이슈 및 고전적인 경영철학분야의 이해 수준 질의
답안작성 전략	개념 문제이므로 최대한 정확한 의미 전달, DBR 시스템에 대한 한정적인 견해가 아닌 폭넓은 지식영역 제시할 경우 고득점 획득가능
참고문헌	The Goal / It's Not Luck / Critical Chain (엘릿 골드렛), TOC (CIC Clobal 자료)
풀이 기술사님	문광석 PE(제 117 회 정보관리기술사 /itpe.moon@gmail.com)

1. 제약 조건 변경 기반 목표지향, 제약이론 TOC(Theory of Constraints)의 개요

개념	엘릿 골드렛 박사가 창시한 시스템의 목적(Goal) 달성을 저해하는 제약 조건을 찾고, 통제하여 전체를 최적화를 달성하는 상식적인 의사결정 지원 경영과학 이론
특징	1) 전체 최적화 : 개별부분의 최적화가 아닌 전체관점의 최적화(지표설정) 2) 제약사항 고려 : 기업의 제약자원을 고려하여 지속적 개선을 추구 3) 집중 개선 : 병목(Bottleneck), 가장 약한 부분이 전체를 좌우

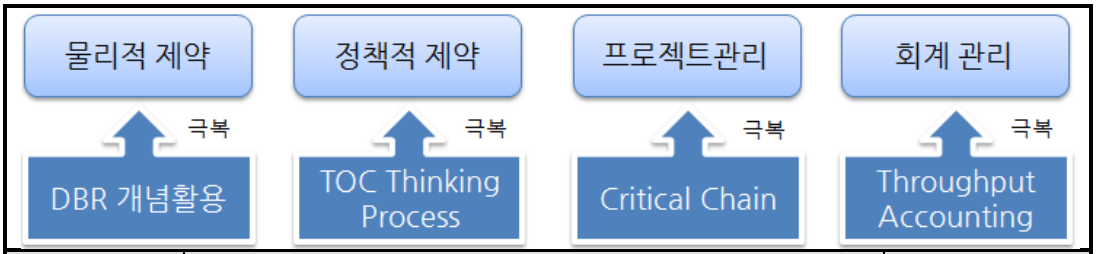
2. TOC 의 문제 해결 절차 및 구성요소

가. TOC 의 문제 해결 절차 (원리)

단계	문제 해결 절차	상세
1 단계	제약 요인 찾기	-시스템의 제약요인(들)을 찾기
2 단계	방법 모색	-제약요인(들)을 최대한 이용할 수 있는 방법을 결정
3 단계	결정 종속	-위의 결정에 다른 모든 것을 종속
4 단계	요인 향상	-시스템의 제약요인(들)을 향상
5 단계	원복	-만약 4 단계에서 제약요인이 더 이상 시스템의 성과를 제약하지 않게 되면 다시 1 단계로 회귀

- 집중개선 프로세스를 통한 원인(사건의 종속성, 통계적 변동), 핵심(부분 최적화 ->전체 최적화)를 통해 문제를 직접적인 해결이 가능
- 다양한 **제약조건 기반 구성요소**별 개별 극복을 통한 다양한 문제 해결이 가능

나. TOC 의 구성요소

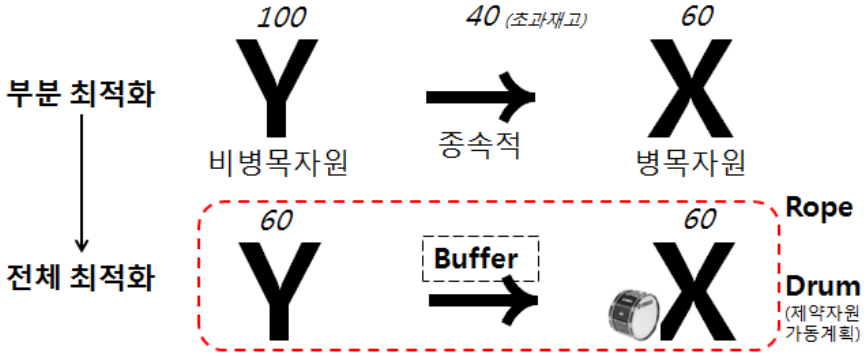


상세구성	설명	활용
DBR (Drum-Buffer-Rope)	- 시스템 제약자원의 소비속도(Drum)에 다른 모든 자원의 소비속도를 맞추고 원자재의 투입은 Drum 에 맞추어 동기화(Rope)시키고 제약자원의 보호를 위한 예비자원(Buffer)를 운용	생산/물류 관리
Throughput Accounting	- 전사적인 TOC 성과 측정 지표(ex. 현금창출율, 재고, 운영비용)	성과측정 지표
Critical Chain	-기존의 CP 개념이 아닌 제약자원관점에서 Buffer 활용하는 CC 적용	프로젝트 관리 (CCM)
TOC Thinking Process	-정책적 제약을 찾아내고 그 제약요인에 변화를 일으키고, 실행으로 옮기는 계통적인 수법	도구 활용

- 제약이론은 학문적인 개념으로 기본원리와 도구, 이를 판단할 지표를 이용하여 **여러 분야에 적용가능**

3. 제약이론 TOC 의 적용 분야 사례

가. 생산/물류 분야의 DBR System



- 부분레벨의 최적화 처리가 아닌 전체 분야에 대한 최적화 처리가 필요

나. 실제 Best Practice 예시

기업	예시 설명
Proctor and Gamble	- 재고 6 억불 감소
Ford 자동차 전자사업부	- 재고 1 억불 감소
Harris 반도체	- 사이클타임 50% 감소, 재고 40% 감소, 수익 28% 향상
Lucent Technology	- 납기 준수율 100%를 달성

- 목표 지향적으로 기존의 제약조건의 한계를 변경하여 의존성 및 사고/성과목표 변경이용

"끝"

3	DOCKER
문제	가상화기술의 도커(DOCKER)
도메인	디지털서비스
정의	오픈소스 프로젝트로, 하이퍼바이저(Hypervisor) 없이 이식 가능한 리눅스 컨테이너(Linux Container, LXC) 기술 바탕, 애플리케이션을 격리된 상태에서 실행하는 경량화 된 가상화 솔루션
키워드	리눅스 컨테이너(LXC), Light Weight, Devops, Libvirt, Namespaces, cgroups, SELinux, 하이퍼바이저, Build, Run, Aufs
출제의도분석	전자정부 표준 프레임워크에서도 지원되고 있으며, 클라우드 서비스(AWS, Oracle 뿐 만이 아닌) MS 윈도우까지도 지원하고 있는 최신 기술
답안작성 전략	예상 문제로써 오랫동안 거론이 되며, 수험자에게 많은 준비가 된 토픽, 전체적인 LXC와 ECO System관점의 개념도와 실무 키워드 위주 차별화
참고문헌	가장 빨리 만나는 도커(이재홍) https://www.slideshare.net/WonchangSong1/docker30-m
풀이 기술사님	문광석 PE(제 117 회 정보관리기술사 /itpe.moon@gmail.com)

1. Immutable Infra & 컨테이너 가상화 기술, 도커(DOCKER)의 개요

가. 도커(DOCKER)의 개념

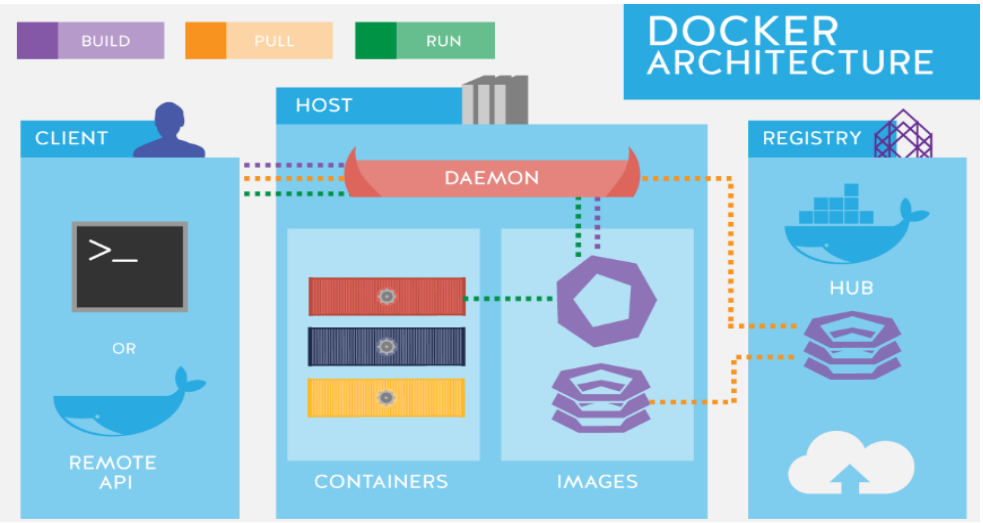
- 오픈소스 프로젝트로, 하이퍼바이저(Hypervisor) 없이 이식 가능한 리눅스 컨테이너(Linux Container, LXC) 기술 바탕, 애플리케이션을 격리된 상태에서 실행하는 경량화 된 가상화 솔루션

나. 도커의 장점

장점	설명
편리한 관리	- 이미지 자체만 관리하면 되어 중앙 관리를 통한 체계적인 배포와 관리 가능
확장	- 이미지 하나로 서버를 계속 생성 가능하여, Auto scaling 기능과 연동
테스트	- 서비스 운영 환경과 동일한 환경으로 구성되어 테스트가 매우 쉬움
경량 인프라	- 운영체제와 서비스 운영환경을 분리하여 가볍고 어디서든 실행 가능

2. 도커(DOCKER)의 Architecture 와 기술 요소

가. 도커의 Architecture



- 이미지는 운영환경 (서버프로그램, 소스, 바이너리등) 묶음으로 구성, 실행방식은 컨테이너로

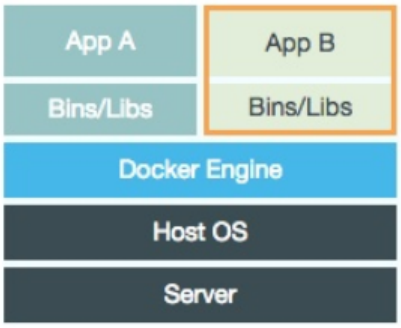
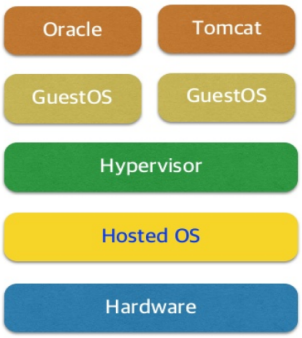
하나의 이미지는 여러개의 컨테이너로 실행 가능

나. 도커의 구성요소

구성 요소	설명	세부 기술
Client	- 도커 컨테이너를 관리하고 실행하기 위해서 데몬과 상호작용하는 Binary 파일	- CLI - Remote API
이미지 (IMAGES)	- 필요한 프로그램과 라이브러리, 소스를 설치한 뒤 파일로 만든 것 - 유니온 파일 시스템 형식	- aufs, btrfs, devicemapper
컨테이너 (Container)	- 이미지를 실행한 상태 - 한개의 이미지로 여러 개의 컨테이너 생성가능	- LXC (namespace, cgroups, SELinux,)
데몬 (daemon)	- Host 에 설치되어 client 와 상호 작용하여 도커 컨테이너를 관리하는 프로세스	- service
레지스트리	- 도커 이미지가 저장되어 있는 장소 - 허브에 등록하여 공유 및 배포하거나 저장소를 직접 만들어 관리	- docker hub - docker registry
네트워크	- 네트워크 터널링을 통해 다른 호스트간의 컨테이너간 데이터 전송	- Network overlay
오케스트레이션	- 멀티 컨테이너를 조직화하고 연결하는 기능 - MSA 어플리케이션의 Fail over, Load balancing	- swarm, mesos kubernates,
모니터링	- 컨테이너 인식 모니터링 - 중앙 관리: 어플리케이션 모니터링, 로그 확인	- cAdvisor, Sensu, Prometheus

- 가벼운 형태의 가상기술로써 기존의 **Hypervisor 기술**과 비견되어 다양하게 클라우드로 이용중

3. 도커와 Hypervisor 기술의 비교

비교	도커	Hypervisor
개념도		
추상화	- OS 커널	- 전체 H/W Device 를 추상화
OS	- 단일 OS 상의 다양한 에디션	- 여러 OS 를 동시 사용
호환성	- Linux 유리, 타 플랫폼은 미비	- 여러 플랫폼 지원
성능	- 직접 OS 를 Access 하여 우수	- Guest/Host OS Layer 로 인해 저하

- 기존의 Hypervisor 기술에 비해 가볍고 빠른 이용을 통해 클라우드 시장의 선두기술로 활용

"끝"

4	블록체인
문제	DID(Decentralized Identity)
도메인	디지털서비스
정의	SSI 구현과 확산을 위해 국제 웹 표준기구인 W3C 주도로 정의된, 개인의 ID 를 중앙 집권으로부터 해방시켜 개인에게 귀속되는 분산형 ID 관리 시스템
키워드	분산 신원 모델, 자기주권 신원지갑, 블록체인, 공인인증서 대체
출제의도분석	최근 DID 세미나(19.6.26)를 비롯해서 블록체인 업계에서 활성화 방안 논의 증가, 118 회 면접 기출문제로써 해당 이슈에 대한 시장의 트렌드 분석 가능 여부 확인
답안작성 전략	정확한 개념에 대한 이해 및 자기주권 신원 지갑과 연계 Insight 제시
참고문헌	https://www.slideshare.net/deframing/blockchain-did-project https://w3c-ccg.github.io/did-spec/
풀이 기술사님	문광석 PE(제 117 회 정보관리기술사 /itpe.moon@gmail.com)

1. 비중앙집권적 ID 관리 시스템, DID(Decentralized Identity)의 개요

가. DID(Decentralized Identity)의 개념

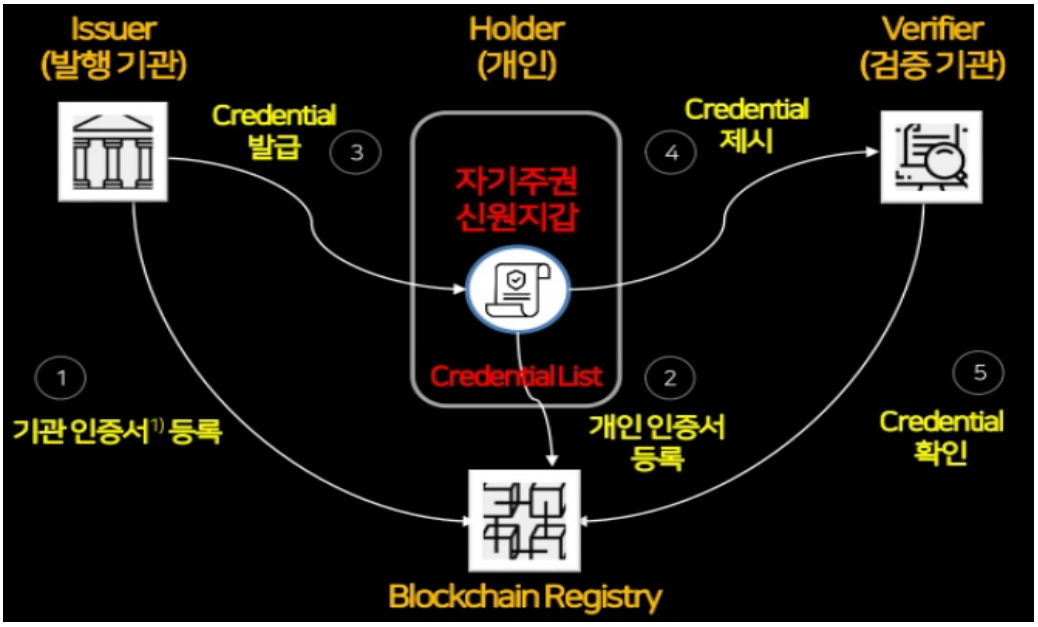
- SSI(자기주권 신원)의 구현과 확산을 위해 국제 웹 표준기구인 W3C 주도로 정의된, 개인의 ID 를 중앙 집권으로부터 해방시켜 개인에게 귀속되는 분산형 ID 관리 시스템

나. DID(Decentralized Identity)의 특징

특징	설명
블록체인 이용	- 블록체인을 PKI(Public Key Infrastructure)로 사용
사용자 관리	- Credential(자격증명 데이터)를 사용자 기기에 저장
제공 내용 검증	- Verifier 가 개인정보 저장하지 않고 고객이 제공한 정보만 검증

2. DID 의 개념도 및 구성 예시

가. DID(Decentralized Identity)의 개념도



- 개인의 Identity 의 증명을 중앙집권적인 관리자 없이 행하는 것이 가능한 블록체인 아키텍처방식

나. DID 의 구성 예시

DID 선언	did:example:123456789abcdefghi
DID 관리	{ "@context": "https://www.w3.org/2019/did/v1", "id": "did:example:123456789abcdefghi", "authentication": [{ // used to authenticate as did:...fghi "id": "did:example:123456789abcdefghi#keys-1", "type": "RsaVerificationKey2018", "controller": "did:example:123456789abcdefghi", "publicKeyPem": "-----BEGIN PUBLIC KEY...END PUBLIC KEY-----₩r₩n" }], "service": [{ // used to retrieve Verifiable Credentials associated with the DID "id":"did:example:123456789abcdefghi#vcs", "type": "VerifiableCredentialService", "serviceEndpoint": "https://example.com/vc/" }] }

- W3C 의 정의를 통한 DID 정의를 통해 직접 관리 가능한 **자기 주권 신원지갑의 구현이** 가능

3. DID 기술을 통한 자기 주권 신원지갑(Self-Sovereign Identity) 구현

- 최근 SKT 텔레콤, KT, LG 유플러스 등 이동통신 3 사가 본인인증 공동 브랜드(PASS)앱 기반 패스 인증서를 출시하여, 본인인증 앱과 연동되는 사설인증서로, 공공기관의 각종 본인확인, 온라인 서류발급 신청 / 금융거래 / 계약서 전자서명 등에 간편하게 이용 가능

"끝"

5	Zero Rating
문제	OTT(Over-the-Top)서비스의 제로 레이팅(zero-rating) 문제점
도메인	디지털서비스
정의	통신사업자가 스폰서하는 제휴 컨텐츠에 대해서 무선 데이터를 과금하지 않는 형태의 자신의 통신서비스의 특화 형태의 요금에 특혜를 주는 서비스
키워드	망중립성, 통신사 게이트키핑, 시장지배력, 플랫폼 중립성,
출제의도분석	5G 상용화 이후에 OTT / 이동통신사의 제로레이팅 증가로 인한 ICT 이슈증대 105 회 컴시응 기출 OTT 와 116 회 정보관리 Zero Rating 의 연계형 문제
답안작성 전략	기 기출문제의 연계형 문제로, 단순 지식 답변이 아닌 자신의 의견 피력
참고문헌	ICT 생태계 변화와 망중립성 정책의 모색: OTT(Over-the-Top) 서비스와 제로 레이팅(Zero-rating) (최은창) 美'제로 레이팅' 허용이 국내 OTT 사업자와 이통사에 미치는 영향 (Mobile Insight)
풀이 기술사님	문광석 PE(제 117 회 정보관리기술사 /itpe.moon@gmail.com)

1. OTT(Over-the-Top)서비스의 제로 레이팅(zero-rating)의 개념

- 통신사업자가 스폰서하는 제휴 컨텐츠에 대해서 무선 데이터를 과금하지 않는 형태의 자신의 통신서비스의 특화된 형태의 요금에 특혜를 주는 서비스

Ex) AT&T 가 디렉 TV 나우에 제로레이팅 적용하여 서비스 제공

2. OTT(Over-the-Top)서비스의 제로 레이팅(zero-rating) 문제점

가. 사업자 측면의 문제점

구분	문제점	상세
서비스 제공자	Platform/Network 경계제거	- 네트워크 망 제공자가 인터넷 상의 Platform 을 통한 디지털 컨텐츠 제공, 클라우드를 통한 서비스 제공으로 경쟁 심화
	시장 경쟁의 해	- 장기적 관점의 특정 서비스 고착화로 이용자들이 피해
	통신사 선택 강요	- 컨텐츠 시장의 경쟁이 왜곡, 통신사 콘텐츠의 게이트 키핑
망제공자	시장 지배력 남용	- 고객의 OTT 서비스 선택시 지정된 이통사로 몰리는 현상
	망중립성 위배	- 망중립성 가이드라인에 저촉 / 위배될 가능성 존재
	통신사 경쟁력	- 무료 데이터 제공이라는 무기를 이용 경쟁력 우위에 이용

- 미 FCC 위원장이 제로 레이팅에 대한 조사 중단을 지시 통해 사실상 허용으로 사업자 유리

나. 이용자 측면의 문제점

구분	문제점	상세
범위	소비자 선택권 훼손	- 사업자가 정해 준 특정 콘텐츠만 이용의 한계
	유선 ISP 로 증가	- 무선영역에서 허용되면, 추후 유선 ISP 로 마찬가지로 답습
서비스	제품 이용 불가	- KT, 삼성 스마트 TV 이용자들의 인터넷 접속 차단 (2012)
	독점에 따른 폐해	- 사용자 -> 서비스 제공자, 서비스 제공자 -> 망제공자에게 최종 비용의 전가를 통한 망제공자의 이익 증가

- 요금의 우위를 주지만, 이용자의 서비스 선택권을 제한, 망중립성 원칙을 위협 문제 존재

3. OTT(Over-the-Top)서비스의 제로 레이팅(zero-rating) 문제점의 해결방안

구분	해결방안	상세
사업자	망중립성 입법	- ICT 생태계 참여자들에게 공평한 기회를 보장
	불공정 모니터링	- 제로 레이팅 건에 불공정행위 발생 가능성 상시 모니터링
	매출 보호를 위한 경쟁관계 서비스 차단 금지	- 자신의 서비스 사용을 위해 경쟁관계에 있는 사용량 제한 불가 (이동통신사의 M-VoIP 의 차단 불가)
이용자	서비스 내용 확인	- 제로 레이팅의 일시적 마케팅 사례 증가로 인한
	원칙 중시	- 일시적인 이득이 아닌 망중립성의 원칙에 대해 중요
	피해 사례 공유	- 불공정 경쟁에 대한 이용자의 피해 사례의 공유 및 고발

- 제로 레이팅으로 인한 서비스 제공시에 넷플릭스와 유튜브와 같은 글로벌 사업자들과의 국내 군소 OTT 기업의 경쟁자체가 불가능하기 때문에 정부의 사후규제가 필수적

"끝"

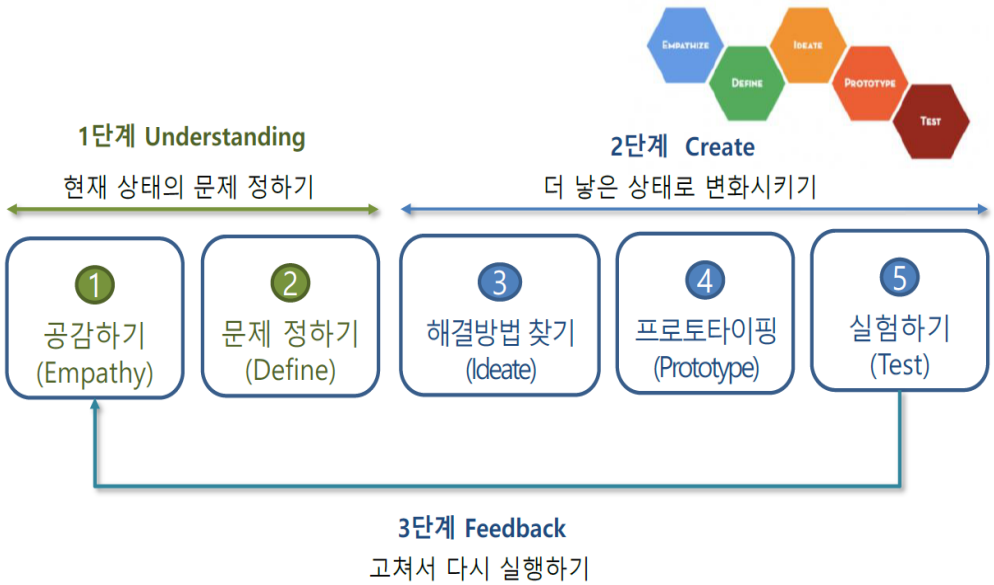
6	Design Thinking
문제	UX 기법인 Design Thinking
도메인	경영
정의	현장과 지속적인 소통을 통해 수요를 파악하며 가능한 모든 대안을 상상·발굴하고 이를 적용·개선하는 방법을 반복함으로써 혁신적 결과를 도출하는 창의적 문제해결 방법
키워드	Empathy(공감), Define(정의), Ideate(상상), Prototype(견본), Test(시험)
출제의도분석	디지털 트윈 같이 용어집에 지속적 언급이 된 사항에 대한 개념 이해 확인
답안작성 전략	Design Thinking에 대한 정확한 개념 및 실 사례 언급
참고문헌	IDEO 의 디자인 씽킹 피플앤인사이트 – Design Thinking Lab NAVER 지식백과 – 창의융합 프로젝트 아이디어 북, 디자인 씽킹이란?
풀이 기술사님	문광석 PE(제 117 회 정보관리기술사 /itpe.moon@gmail.com)

1. 창의적 문제해결 방법, UX 기법인 Design Thinking 의 개요

정의	현장과 지속적인 소통을 통해 수요를 파악하며 가능한 모든 대안을 상상·발굴하고 이를 적용·개선하는 방법을 반복하여 혁신적 결과를 도출하는 창의적 문제해결 방법
특징	1) 고객 중심주의(Human Centered Design) : 인간 중심 접근법으로 고객 및 사용자에 대한 공감 및 감정이입 통하여 문제 해결의 단서를 찾는 것 2) 신속한 프로토타입 : 프로토타입 제작 통해 해당 아이디어의 강점과 약점을 배우고 개선을 위한 새로운 방향성을 설정하기 위한 과정

2. Design Thinking 5 단계 프로세스 및 상세 설명

가. Design Thinking 의 5 단계 프로세스



- 현재 상태의 문제공감에서부터 개선하고 그 결과를 피드백하면서 최종 결과를 도출하는 디자인 씽킹 5 단계

나. Design Thinking 의 5 단계 프로세스 상세 설명

프로세스	설명	적용방법론
공감하기 (Empathy)	- 사람들을 유심히 관찰하고 인터뷰하면서 공감하는 과정 - 상황관찰 및 문제점 발견	설문조사, 인터뷰, 관찰
문제정의 (Define)	- '공감' 한 결과를 확인하여 무엇이 문제인지 정의를 내리는 단계 - 문제의 인식 및 공유	요구 사항 정의, 의견 조율, 인사이드 매트릭스
아이디어 도출 (Ideate)	- 문제점으로 파악된 것에 대한 해결방법을 만드는 과정 - 아이디어 확산 및 발산	브레인스토밍, 아이디어스케치, 시각화
프로토타이핑 (Prototype)	- 도출된 아이디어로 프로토타이핑을 만드는 과정 - 아이디어 구현하기	프로토타이핑, 3D 프린팅
테스트 (Test)	- 만들어진 프로토타이핑 테스트 - 피드백을 통한 아이디어 개선 및 회고	사용성테스트, 반복적 테스트

- **Business Thinking** 과 더불어 다양한 방식으로 해답을 만들고 검증하기 위해서 이용

3. 디자인 씽킹(Design Thinking)과 비즈니스 씽킹(Business Thinking)의 비교

비교	디자인 싱킹	비즈니스 싱킹
기본 가정	- 주관적 경험 - 실현은 사회적으로 구성	- 합리성, 객관성 - 현실은 고정적, 양적 측정 가능
방법론의 목표	- 더 나은 해답을 향해 반복 실험	- 최고의 하나의 해답을 입증
과정	- 실행하기	- 계획 세우기
의사결정 동인	- 감정적 직관, 실험 모형	- 논리, 수치 모형
가치	- 참신함을 추구 - 그대로 머물러 있는 상태를 거부	- 통제와 안전성을 추구 - 불확실성을 불편
관심의 수준	- 추상적 단계와 매우 구체적 단계 사이에서 반복적으로 이동	- 추상적 또는 매우 구체적

- 디자인 씽킹과 비지니스 씽킹은 서로 반대되거나 양극단에 있는 것이 아니며, 하나의 문제를 해결하는데 서로 다른 관점 / 접근 방법을 제공하여 이용

"끝"

[기타자료] 사례

기업	사례	설명
IDEO	'Keep the change' 직불카드	- 직불카드 결제 시 정수로 계산하고 남은 잔돈 계좌 이체하는 방식
인튜이트 (Intuit)	연말정산 프로그램 차트 속 '자주하는 질문'	- 자주하는 질문을 연말 정산 프로그램 차트에 보여주는 기능 추가
현대카드	카드 청구서 '총사용금액'	- 카드청구서가 세로로 나오는데 보기 불편하다는 불만을 총 사용금액을 큰글씨로 맨 위에 표기한 새 청구서 제작
스탠포드 D-스쿨	임브레이스 인펀트 워머 (Embrace Infant Warmer)'	- 인큐베이터가 부족해 미숙아 사망률이 높은점을 해결하기 위해 최소한의 체온 유지를 지원하는 200 달러 저비용 미니 어처 파우치 보급

▶ **포드의 하이브리드 자동차, 퓨전의 계기판 인터페이스**

포드사에서 '퓨전(Fusion)'이라는 하이브리드 자동차를 개발할 때, IDEO에 의뢰가 들어왔다. 사용자들이 하이브리드 자동차를 잘 이용할 수 있는 인터페이스로 만들어달라는 요구였다. 그래서 IDEO측 사람들은 현장에 나가 사람들이 차의 인터페이스와 어떻게 상호작용하는지, 또 하이브리드 자동차가 움직이는 방식을 어떻게 이해하는지 등에 대해서 연구했다.

이들은 흥미로운 사실을 발견했는데, 하이브리드 자동차를 구입하는 사람들 중에 환경에 관심 있는 사람들이 많다는 것이었다. 연구자들은 그 관심을 자동차와 상호작용하는 방식에 적용했다.

다음 중 첫 번째 그림은 시동 전 계기판 모습이다. 운전자가 경제속도로 운전하면, 계기판은 아래 두 번째 그림과 같이 파란 잎사귀가 자라는 모습을 디스플레이한다. 이는 바로 환경에 관심이 많은 소비자를 고려한 디자인으로, 경제속도로 운전함으로써 운전자가 환경에 기여한다는 느낌을 주어 제품에 대한 만족을 느끼게 한다.

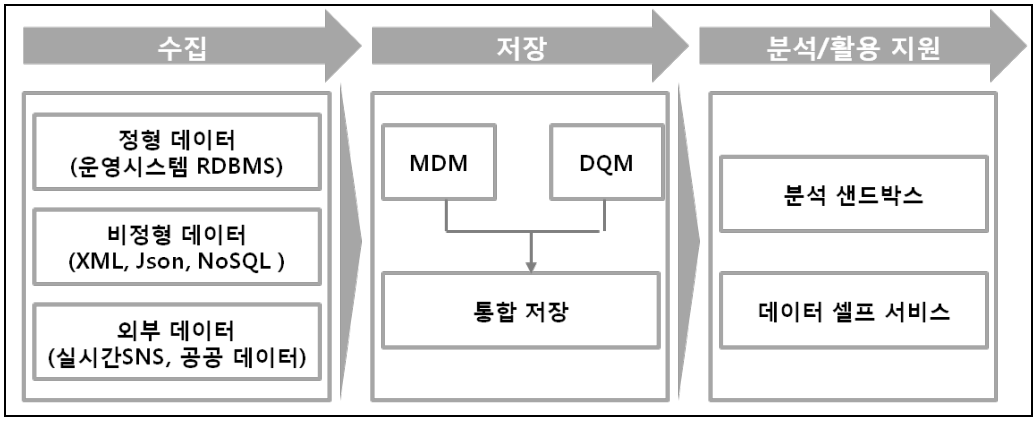
7	Data Lake
문제	Data Lake
도메인	Big Data
정의	조직 내/외부 다양한 원천시스템으로부터 수집된 데이터의 크기, 유형, 속도가 상이한 다양한 데이터를 수집하고 원하는 형태로 활용을 지원하는 빅데이터 저장소
키워드	통합 저장, 확장성, 일관성 , 데이터 늪(데이터 품질 관리), 분석 샌드박스, 셀프 서비스
출제의도분석	빅데이터 활용도 향상을 위해 구축하는 Data Lake 에 대한 관심 증가
답안작성 전략	Data Lake 의 개념과 구성요소를 상세히 작성 Data Lake 의 늪지 방지를 위한 고려 사항 등 추가 의견 제시
참고문헌	http://www.itfind.or.kr/WZIN/jugidong/1861/file21047232475214105 03-186102.pdf 데이터 레이크 기술 동향과 도입원칙 , 백현
풀이 기술사님	서봉국 PE(제 118 회 정보관리기술사 / 79euros@gmail.com)

1. 진정한 빅데이터 활용을 위한 데이터 저장소, Data Lake 의 개념

- 조직 내/외부 다양한 원천시스템으로부터 수집된 데이터의 크기, 유형, 속도가 상이한 다양한 데이터를 수집하고 원하는 형태로 활용을 지원하는 빅데이터 저장소

2. Data Lake 구성도 및 구성요소

가. Data Lake 의 구성도



- 다양한 내 외부 데이터의 수집, 관리 및 통합 저장, 분석 지원의 형태로 구성

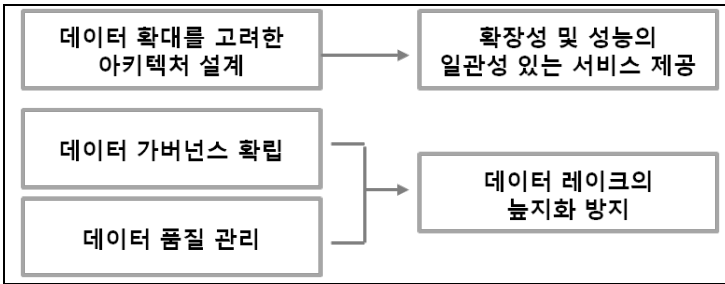
나. Data Lake 의 구성요소

구분	요소	설명	기술
데이터 수집	정형데이터	조직 내 RDBMS 기반 시스템	Apache Nifi , Flume
	비정형데이터	조직 내/외부 Text , Image, Audio Data	

데이터 관리	데이터 품질 관리	정합성, 견고성, 유일성 등 기준으로한 데이터 품질 관리	DB Profiling DB Cleansing
	Master Data 관리	Master Data 통합, 표준화, 관리, 모니터링	MDM
데이터 저장	데이터 통합 저장	통합 분산 빅데이터 저장 기술	HDFS
데이터 활용 지원	분석 샌드박스	원본 데이터의 보호 및 Data Scientist 에게 격리된 분석 환경 제공	AI 및 시각화 지원 기술 (Tensor Flow, D3.js, Tableau)
	데이터 셀프 서비스	현업 사용자의 데이터 분석/활용 환경 제공	

- 데이터 수집 및 품질관리 단계를 거쳐 저장한 통합 데이터를 분석 및 시각화에 활용

3. Data Lake 성공적 구축/활용을 위한 고려사항



- 높은 데이터 품질 기반 일관성 있는 서비스 제공

- 다크 데이터 발굴, Insight 도출 등 가시적 빅데이터 활용 성과 창출 기여

"끝"

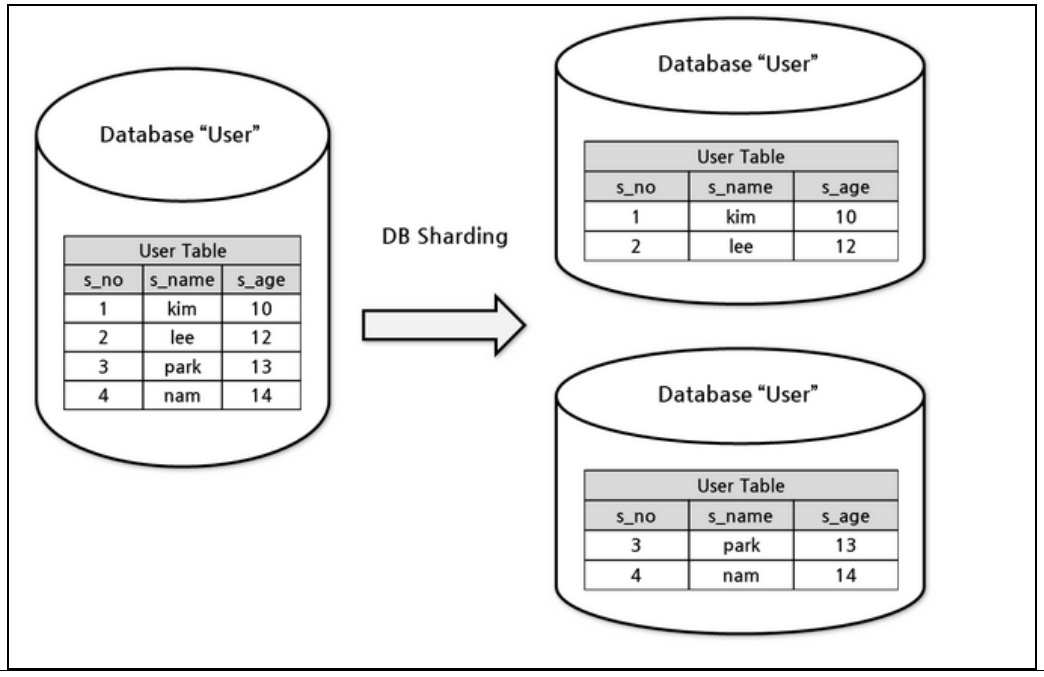
8	Database Shard
문제	Database Shard
도메인	Database
정의	물리적으로 다른 데이터베이스를 수평 분할 방식으로 분산 저장/조회하는 기법
키워드	수평분할, 물리적 DB, 위치 추상화 Shard metadata, key
출제의도분석	빅데이터의 증가에 따른 성능 향상 기법 중 하나로 몽고 DB 등에 사용되는 Shard 에 대한 개념 숙지 필요
답안작성 전략	Shard 에 대한 정확한 개념 및 구성요소 기술 Shard 분할 방법 또는 파티셔닝 등 유사 성능향상 기법과의 비교
참고문헌	2018.11 KPC 모의고사 관리 기출 풀이, 서브 노트
풀이 기술사님	서봉국 PE(제 118 회 정보관리기술사 / 79euros@gmail.com)

1. 데이터 베이스 성능향상을 위한 Database Shard 개념 및 특징

개념	특징
- 물리적으로 다른 데이터베이스를 수평 분할 방식으로 분산 저장/사용하는 방법 - 하나의 데이터베이스를 Shard 라 부르는 각각의 개별 파티션으로 분할하는 기법	- 데이터 분할 - 성능 개선 - 높은 신뢰성 - 위치 추상화

2. Database Shard 개념도/구성요소 및 분할 방법

가. Database Shard 개념도 및 구성요소



구분	요소	설명
시스템 구성요소	Shard DB	분할 데이터 저장, 사용자 요청 처리 물리 DB
	Proxy	Shard metadata 활용 실제 질의 처리를 Shard DB 로 전달하는 역할을 하는 미들웨어 프로세스
데이터 구성요소	Shard metadata	Shard 동작을 위한 설정 정보 , 요청된 질의 기반 Shard 선택을 위한 정보 및 세션 생성 정보 포함
	Shard Key	Sharding 된 테이블에서 Shard 를 선택하기 위한 식별자로 사용되는 칼럼
	Shard ID	Shard Database 식벽자

나. Database Shard 분할 방법

방법	설명	특징
Vertical Partitioning	테이블별로 서버를 분할하는 방식 각 서버 데이터 거대화시 추가 샤딩 필요	구현 간단, 시스템 구조 변경 최소화
Range Based Partitioning	하나의 Feature 나 Table 거대화 시 샤딩	데이터 분할 방법 예측 가능 시 구현 가능
Key or Hash Based Partitioning	엔티티를 해쉬 함수에 넣어 나오는 값 이용 서버를 결정하는 방식	결과 값이 균등 분포되게 해쉬 함수 설계 필요
Directory Based Partitioning	파티셔닝 메커니즘 제공 추상화 서비스 생성	샤드 키 Look-up 기능 구현

- 데이터의 성격과 특징에 따라 Table 단위를 작게 유지할 수 잇는 Shard 분할 방법을 결정해야 하며 적용 시 서비스 정지 없이 데이터 재분배 수행 필요

"끝"

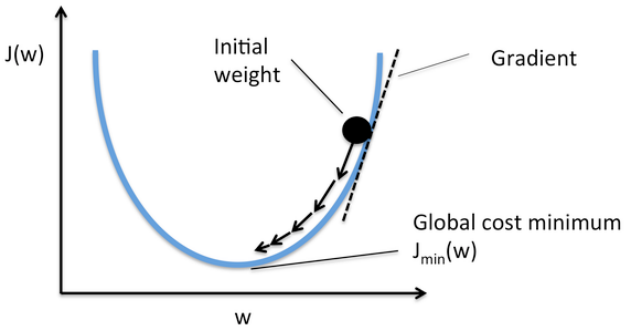
9	기계학습의 Gradient Descent Algorithm
문제	기계학습의 Gradient Descent Algorithm
도메인	AI
정의	주어진 함수의 최소값 위치를 찾기 위해 비용 함수(Cost Function)의 그레디언트 반대 방향으로 정의한 step size 를 가지고 조금씩 움직여 가면서 최적의 파라미터를 찾으려는 방법
키워드	경사하강법, Cost Function, Step Size(or Learning Rate), Global cost minimum,
출제의도분석	머신러닝, AI 에서 가장 기본이 되는 최적화 기법에 대한 이해도 확인
답안작성 전략	Gradient Descent 의 사용 목적과 알고리즘 동작 구조 설명 Gradient Descent 문제점 해결을 위한 고급 알고리즘 제시
참고문헌	"모두의 딥러닝 : 원리를 쉽게 이해하고 나만의 딥러닝 모델을 만들 수 있다 with 텐서플로(tensorflow) & 케라스(keras)", 조태호
풀이 기술사님	서봉국 PE(제 118 회 정보관리기술사 / 79euros@gmail.com)

1. 최적해를 찾기 위한 Gradient Descent Algorithm 개념

- 주어진 함수의 최소값 위치를 찾기 위해 비용 함수(Cost Function)의 Gradient 반대 방향으로 정의한 step size 를 가지고 조금씩 움직여 가면서 최적의 파라미터를 찾으려는 방법
- 머신 러닝 기법 등에서 오차항의 최소값을 찾기 위해 사용되는 기법

2. Gradient Descent Algorithm 의 메커니즘 및 분류

가. Gradient Descent Algorithm 의 메커니즘



Formal definition

$$cost(W) = \frac{1}{2m}\sum_{i=1}^{m}(Wx^{(i)} - y^{(i)})^2$$

$$W := W - \alpha\frac{\partial}{\partial W}cost(W)$$

1) 임의의 변수 초기값 선택
2) 변수 값에 해당하는 경사도(기울기) 계산
3) 변수를 경사방향으로 움직여 다음 기울기 계산
4) 1),2),3)을 반복하여 함수의 최소값이 되는 값을 찾음

- 머신 러닝에서 예측 값과 실제 값의 오차를 설명하는 오차함수의 최소값을 찾기 위해 사용
- 업데이트 시마다 매번 전체 데이터 미분으로 인한 성능 문제 해결과 최적 값의 정확성 향상을 위해 다양한 경사하강법 등장

나. Gradient Descent Algorithm 분류 및 설명

- 정확성과 효율(성능)을 높이기 위한 고급 경사하강법 알고리즘 발전

Gradient Descent Algorithm	설명	효과
SGD (Stochastic Gradient Descent)	랜덤하게 추출한 일부데이터만 이용 ➔ 빠른 속도로 최적해의 근사값 계산	속도 개선
Momentum	관성의 방향을 고려해 진동과 폭을 줄이는 효과	정확도 개선
네스테로프 모멘텀 (NAG)	모멘텀이 이동시킬 방향으로 미리 이동해서 그레이디언트를 계산. 불필요한 이동을 줄이는 효과	정확도 개선
아다그라드(Adagrad)	변수의 업데이트가 잦으면 학습률을 적게 하여 이동 보폭을 조절하는 방법	보폭 크기 개선
알엠에스프롭(RMSProp)	아다그라드의 보폭 민감도를 보완한 방법	보폭 크기 개선
아담(Adam)	모멘텀과 알엠에스프롭 방법을 합친 방법	정확도, 보폭 크기 개선

"끝"

10	Lehman 의 SW 변화 원리
문제	Lehman 의 SW 변화원리
도메인	소프트웨어 공학
정의	소프트웨어의 지속적 진화에 대해 SW 를 3 개의 Type 으로 나누고 Software Evolution 에 대해 설명한 8 가지 원리
키워드	계속적 변경, 복잡도 증가, 자기규제, 안정화, 친근성, 지속적 성장, 감소하는 품질, Feedback System,
출제의도분석	119 회 시험의 대표적인 기본 토픽 문제로 소프트웨어 공학의 기본 개념 숙지 확인
답안작성 전략	정확한 8 가지 원리 설명, 적용 방안 제시
참고문헌	KPC 2016 년 7 월 합숙, 서브노트
풀이 기술사님	서봉국 PE(제 118 회 정보관리기술사 / 79euros@gmail.com)

1. 리만의 SW 변화원리

- 소프트웨어의 지속적 진화에 대해 SW 를 3 개의 Type 으로 나누고 Software Evolution 에 대해 설명한 8 가지 원리

2. 리만의 소프트웨어 변화 원리 상세 설명

가. 리만의 소프트웨어 분류

구분	관점	내용
S-Type(Static)	Specification and Solutions	정해진 명세에 따라 정확히 동작하는 SW
P-Type(Practical)	Procedures	절차나 실행 입력값에 따라 다르게 수행되는 SW
E-Type(Embedded)	Environment	실 세계 환경적 상황과 밀접하게 연관되어 동작하는 SW

- 리만은 위의 분류 중 E-Type 에 대한 변화 원리를 제시

나. 리만의 소프트웨어 변화 원리 상세 설명

No	원리	설명
①	계속적 변경	소프트웨어는 계속 진화하며 요구사항에 의해 계속 변경
②	복잡도증가	변경이 가해질수록 구조는 복잡해짐
③	자기 규제	각 단계마다 진화과정이 피드백되어 자가 규제를 적용
④	조직적 안정화 유지	작업량의 불변을 의미
⑤	친근성 유지	SW 각버전의 변화는 일정하며 규칙적 수행결과 및 추이를 보여주므로 예측 가능
⑥	Continuing	SW 의 기능적 내용은 사용자 만족도 유지를 위해 Lifetime

	Growth	동안 지속적 성장
⑦	감소하는 품질	SW 는 엄격하게 관리 및 운영되지 않거나 , 환경변화에 적응하지 않는다면 품질은 감소함
⑧	Feedback System	중요한 SW 제품을 개선하기 위해 피드백 시스템으로 구성

- 리만의 소프트웨어 변화 원리는 유지보수 분류에 적용 가능

3. 리만의 소프트웨어 변화 원리의 유지보수 적용 방안

구분	세부내용	적용원리
조직 (People)	유지보수/변경관리를 위한 전문화 조직 구성	조직적 안정화 유지
조직적 안정화 유지	요구사항 및 변경에 대한 통제프로세스 구성	Continuing Growth 복잡도증가
시스템 (System)	프로그램과 버전에 대한 시스템적 구성 Baseline/CMDB 구성, 형상관리시스템 구성	친근성 유지

- 복잡도 증가의 원리에 따라 변경을 수행 후 회귀테스트(Regression Test)를 수행하여 Unit, Input Partition, Path, Data flow 에 대한 검증을 통하여 복잡성 증가에 대한 품질 보증 활동 수행이 요구됨

"끝"

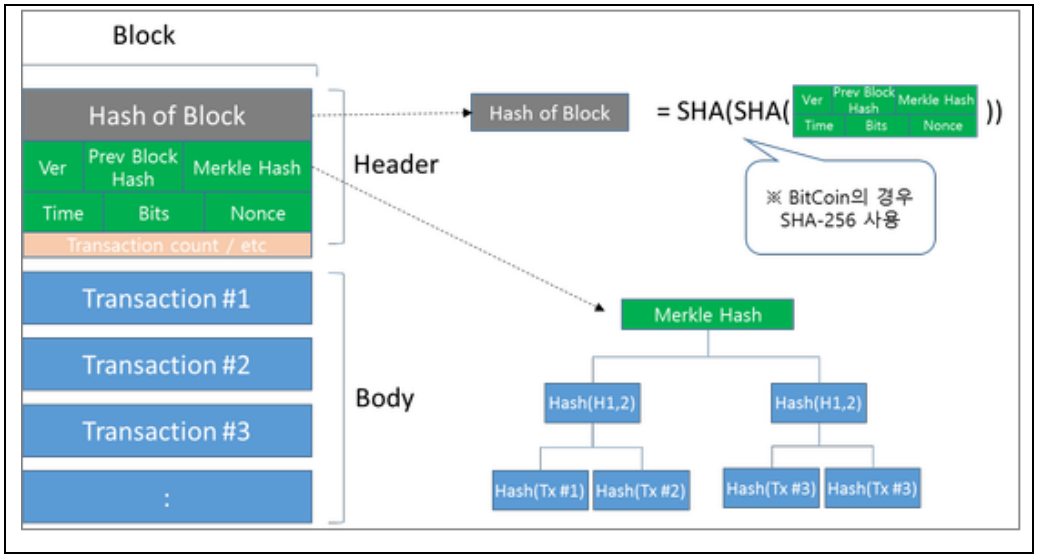
11	머클트리(Merkle Tree)
문제	머클트리(Merkle Tree)
도메인	Blockchain
정의	해시트리의 일종으로 RalphMerkle 고안이 고안한 모든 비 리프노드의 이름이 자식노드들의 해시로 구성된 트리
키워드	Merkle Root , Merkle Path, SHA-256, Full/Light Node
출제의도분석	119 회 전반에 걸쳐 블록체인은 전체적인 전략 문제와 세세한 기술부분까지 함께 기출, 본 문제는 블록체인을 구성하는 기본 기술에 대한 질문
답안작성 전략	머클트리 기술에 대한 상세 기술, 블록체인에서의 사용 목적 및 장점
참고문헌	http://wiki.hash.kr/index.php/%EB%A8%B8%ED%81%B4%ED%8A%B8%EB%A6%AC , 해시넷
풀이 기술사님	서봉국 PE(제 118 회 정보관리기술사 / 79euros@gmail.com)

1. 블록체인 인증을 위한 핵심 기술, 머클 트리의 개념

- 해시트리의 일종으로 RalphMerkle 고안이 고안한 모든 비 리프노드의 이름이 자식노드들의 해시로 구성된 트리

2. 머클 트리의 구성 및 원리

가. 머클트리의 구성



- 트랜잭션의 위/변조 검증을 빠르게 하기 위한 구조로, 특정 노드의 비교만으로 트랜젝션 위/변조 검출 가능

나. 머클트리의 구성요소

구분	요소	설명
머클트리 구성요소	Merkle Root	- 모든 거래의 요약본 - 블록이 보유하고 있는 거래 내역들의 해시값을 가장 가까운 거래내역끼리 쌍을 지어 해시화하고 쌍을 지을 수 없을 때까지 이 과정을 반복했을 때 얻게 되는 값

	Merkle path	- 어떤 거래의 진위를 따질 때 이를 검증하는 과정
	Full Node	- 제네시스 블록부터 현재 시점의 형성된 블록이 연결된 블록체인 전체를 유지하는 노드
	Light Node	- 일부 블록만 소유하고 Full Node 에게서 필요한 정보만을 받아서 유지하는 노드
암호화	SHA-256	- 머클트리가 데이터를 간편하고 확실하게 인증하기 위해 사용하는 해시함수 기반 단방향 암호화 기술 - 거래량의 상관없이 SHA-256 를 사용하여 작은 용량으로 유지

- 머클트리 자체가 해시로 이루어진만큼 하나의 트랜잭션 혹은 블록 내 필드값이 변조될 경우 머클루트 해시 값이 변조되는 쇄도 효과(avalanche effect)가 발생하여 위/변조 방지

3. 머클 트리의 활용

활용	설명
트랜잭션 변경 확인	블록체인 거래 발생 시 트랜잭션의 정보들이 변경되었는지 확인
트랜잭션 유효성 보장	머클루트만 헤더에 담아 트랜잭션의 유효성 보장
Merkle path 제공	Merkle path 제공으로 트랜잭션의 블록이 유효한지 효율적으로 검사 가능

- 대표적으로 비트 코인에서 사용되며 이더리움의 경우 패트리샤 트리(Patricia Tree) 사용

"끝"

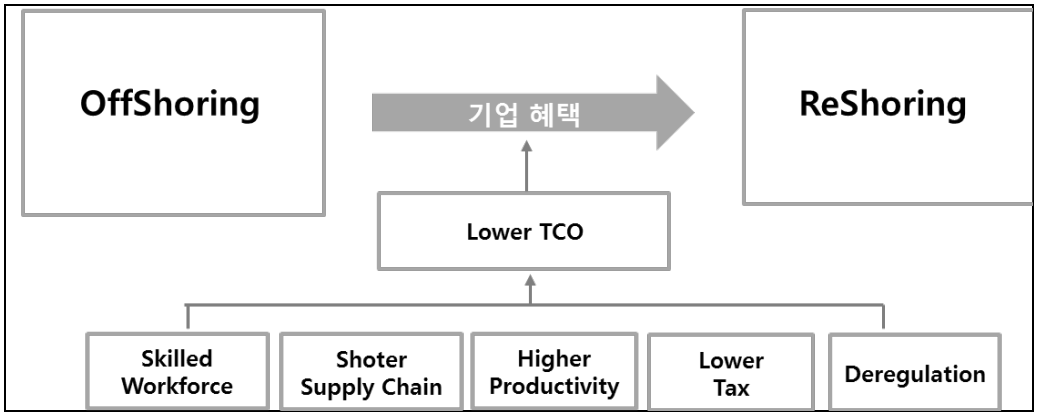
12	리쇼링(Reshoring)
문제	리쇼링(Reshoring)
도메인	경영
정의	인건비, 비용 등의 이유로 해외에 나가 있는 자국 기업을 각종 세제 혜택과 규제 완화 등을 통해 불러 들이는 정책
키워드	세제 혜택, 규제완화, 스마트 팩토리 Lower TCO, Shoter Supply Chain, Higher Productivity
출제의도분석	미국을 필두로 시도되고 있는 자국 우선주의의 정책 중 하나인 리쇼어링 유도에 대한 개념과 기업이 이를 선택하게 되는 실현요소에 대한 이해 요구
답안작성 전략	리쇼어링의 개념과 기업의 실현요소와 인건비 측면의 이득을 위해 기업들이 수행해온 오프쇼링과의 비교
참고문헌	https://nrc.re.kr/boardFileDown.do?file_idx=108657 주요국 리쇼어링 정책의 전개와 시사점
풀이 기술사님	서봉국 PE(제 118 회 정보관리기술사 / 79euros@gmail.com)

1. 리쇼링(Reshoring)의 정의

- 인건비, 비용 등의 이유로 해외에 나가 있는 자국 기업을 각종 세제 혜택과 규제 완화 등을 통해 불러들이는 정책

2. 리쇼링의 개념도 및 실현 요소

가. 리쇼링의 개념도



- 리쇼링의 다양한 실현 요소들로 TCO 를 낮추고 기업 혜택을 늘림으로써 기업들의 Reshoring 을 유도

나. 리쇼링의 실현요소

구분	요소	설명
생산성 향상	Skilled Workforce	고급 인력의 활용을 통한 기술 개발 및 생산성 향상
	Higher Productivity	스마트 팩토리 활용, 생산자동화를 통한 낮은 노동력 투입 고 생산성 획득
비용	Shorter Supply Chain	공급망 간 거리를 줄임으로써 공급 비용 절감

절감	Lower Tax	세금 감면/할인 혜택을 통한 영업이익 증가
사업성 향상	Deregulation	규제 완화로 인한 신사업 발굴 기회 확대

- 생산성 향상, 비용절감, 사업성 향상의 실행요소 기반 Reshoring

3. 리쇼링과 오프쇼링 비교

구분	리쇼어링	오프쇼어링
개념	단순한 공급망, 세제 혜택, 높은 인력 수준을 기반으로 제조거점을 본국에 다시 이전하는 정책	기업 업무의 일부를 저임금의 해외 기업에 맡기는 아웃소싱 형태의 정책
목적	제품과 서비스의 결합 기반, 생산 비용 대비 높은 품질의 제품 적기 출시	저원가 제조환경의 구축
산업 구조	스마트 팩토리 활용 등 첨단 융합 산업구조	노동 집약적 제조 중심 산업 구조
확산 배경	제조업의 서비스화, ICT 기술 발전, 자국 우선주의	적은 인건비 및 제조원가/비용

- 선진국의 리쇼링 전략의 핵심은 ICT 기술에 기반한 스마트 팩토리이며 고도화되고 기술 집약적인 스마트 팩토리를 구축하여 시장경쟁 우위의 제품의 품질, 생산 효율성을 극대화하고자 하는 시도가 늘고 있음

"끝"

<참고 : 주요국 리쇼어링 정책>

국가	정책
독일	인더스트리 4.0 정책을 통해 스마트 공장, 클라우드 컴퓨팅, 가상-물리 시스템 통합 등 혁신적 신기술을 활용하고자 하는 기업의 리쇼어링을 장려
미국	민간 기업 주도의 스마트 팩토리 확산, 법인세 38% → 21%, 해외 생산품 미국 수출시 35% 관세 부과, 공장이전비 20% 지원 등
대만	해외로 진출했던 기업의 복귀를 장려하기위해 세금감면, 토지제공, 자금지원과 함께 R&D 지원 정책 시행
한국	2013 년부터 「해외진출기업복귀법」을 제정하여 유턴기업에 대해 세금 혜택, 보조금, 인력 및 입지 지원 등을 제공하고 있지만 혁신 및 R&D 관련 지원은 없는 실정

13	Database Isolation Level
문제	Database Isolation Level
도메인	Database
정의	병행 트랜젝션 실행 시 일관성 있게 데이터를 읽을 수 있도록 데이터를 허용하는 수준
키워드	Read Uncommitted, Read Committed, Repeatable Read, Serializable , 동시성, 일관성
출제의도분석	출제의도 입력
답안작성 전략	Isolation Level 에 대한 상세 개념 작성 및 이상현상과의 관계 작성 Database 동시성과 일관성의 Trade Off 관계 설명
참고문헌	2019 년 1 월 KPC 합숙 ,
풀이 기술사님	서봉국 PE(제 118 회 정보관리기술사 / 79euros@gmail.com)

1. 데이터 베이스 고립성 허용 수준, Database Isolation Level 의 개념

- 병행 트랜젝션 실행 시 일관성 있게 데이터를 읽을 수 있도록 데이터를 허용하는 수준

2. Database Isolation Level 설명 및 단계 별 발생 가능 이상현상

가. Database Isolation Level 설명

Isolation Level	설명	특징
Read Uncommitted (Level 0)	아직 커밋되지 않은 데이터를 다른 트랜젝션에서 읽는 것을 허용	오라클 지원 안함
Read Committed (Level 1)	트랜잭션이 커밋되어 확정된 데이터만 읽는 것을 허용	S-Lock 설정
Repeatable Read (Level 2)	자신의 트랜젝션이 끝날 때까지 다른 트랜젝션이 Update 또는 Delete 할 수 없도록 함	S-Lock, X-Lock 설정
Serializable	고립 수준이 가장 높은 레벨로 타 트렌잭션의 Insert 금지	Index S-Lock 설정, Undo 데이터 사용 구현

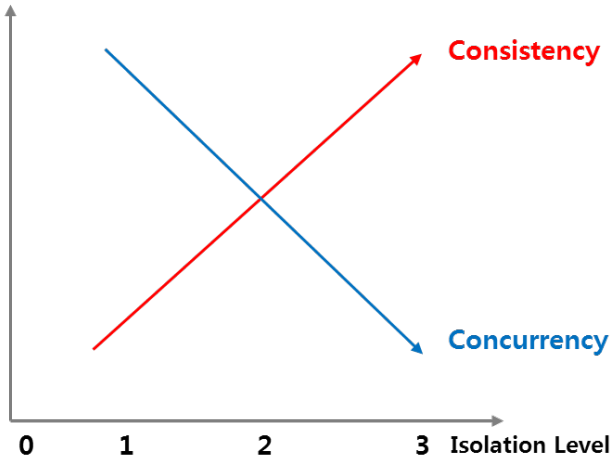
- 각 Isolation level 에 따라 DB 이상현상이 발생

나. Database Isolation Level 단계 별 발생 가능 이상현상

Isolation Level	Dirty Read	Non-Repeatable Read	Phantom Read
Read Uncommitted	발생 가능	발생 가능	발생 가능
Read Committed	발생 불가	발생 가능	발생 가능
Repeatable Read	발생 불가	발생 불가	발생 가능
Serializable	발생 불가	발생 불가	발생 불가

- DB 이상현상을 해결하기 위해서는 Isolation Level 을 올려야 하지만 이 경우 동시성과 일관성의 Trade Off 가 발생

3. Database Isolation Level 과 동시성/일관성의 Trade off



설명
Isolation Level 상승 시 일관성 향상되나 Lock 이 수행되어 동시성은 저하
동시성과 일관성의 Trade Off 를 고려한 적절한 Isolation Level 설정 필요

"끝"