발간등록번호
11-1741050-000038-01

연구결과보고서

데이터세트 유형 전자기록의 장기보존기술 연구

Study on long-term preservation technology of dataset-type electronic records

주관연구기관: 전북대학교 산학협력단

국가기록원

용역연구과제 최종보고서

사 업 명	2019년 국가기록관리· 활용기술 연구개발 사업			
과 제 명	국문	데이터세트 유형 전자기록의 장기보존기술 연구		
	영문	Study on long-term preservation technology of dataset-type electronic records		
주관연구기관	기 관 명		소재지	대 표
	전북대학교 산학협력단		전라북도 전주시 덕진구 백제대로567	조 재 영
주관연구 책임자	성 명		소속 및 부서	전 공
	양 동 민		전북대학교 기록관리학과 문화융복합아카이빙 연구소	컴퓨터공학
총연구기간	2019년 5월 2일 - 2019년 11월 29일(7개월)			
총 연구비	166,000천원			
연구년차	연구기간		연구비	
1차년도	2019.05.02 - 2019.11.29		166,000천원	
총참여연구원	15명 (책임연구원: 2명, 연구원: 9명, 연구보조원: 4명)			

2019년도 용역연구개발사업에 의하여 수행중인 연구과제의 연구결과보고서를 붙임과 같이 제출합니다.

붙임 : 1. 연구결과보고서 15부.

2019년 11월 28일

주관연구책임자 양 동 민 (인 또는 서명)

주관연구기관장 조 재 영 (직인)

국가기록원장 귀하

제 출 문

국가기록원장 귀 하

이 보고서를 “데이터세트 유형 전자기록의 장기보존기술 연구(전북대학교 산학협력단/양동민)” 과제의 연구결과보고서로 제출합니다.

2019. 11. 28.

주관연구기관명 : 전북대학교 산학협력단

주관연구책임자 : 양 동 민

제 1세부과제명 : 데이터세트 유형 전자기록 보존포맷 선정 및 테스트베드 구축・시험・검증
(제1세부연구기관/세부과제책임자): 전북대학교 산학협력단/양동민

제 2세부과제명 : 클라우드 기반 전자기록의 장기보존기술개발 테스트베드 구축 및 에뮬레이션 시험·검증
(제2세부연구기관/세부과제책임자): ㈜이노그리드/조철용 실장

목 차

Ⅰ. 연구개발결과 요약문
(한글) 데이터세트 유형 전자기록의 장기보존기술 연구
(영문) Study on long-term preservation technology of dataset-type electronic records

Ⅱ. 총괄연구개발과제 연구결과
제1장 총괄연구개발과제의 최종 연구개발 목표 ····· 9P
제2장 총괄연구개발과제의 최종 연구개발 내용 및 방법 ····· 37P
제3장 총괄연구개발과제의 최종 연구개발 결과 ····· 41P
제4장 총괄연구개발과제의 연구결과 고찰 및 결론 ····· 45P
제5장 총괄연구개발과제의 연구성과 ····· 49P
제6장 기타 중요변경사항 ····· 52P
제7장 참고문헌 ····· 53P
제8장 첨부 및 별첨 서류 목록 ····· 57P

Ⅱ. 제1세부연구개발과제 연구결과
제1장 제1세부연구개발과제의 최종 연구개발 목표 ····· 59P
제2장 제1세부연구개발과제의 연구개발 내용 및 방법 ····· 68P
제3장 제1세부연구개발과제의 최종 연구개발 결과 ····· 74P
제4장 제1세부연구개발과제의 연구결과 고찰 및 결론 ····· 203P
제5장 제1세부연구개발과제의 연구성과 ····· 207P
제6장 기타 중요변경사항 ····· 209P
제7장 참고문헌 ····· 210P
제8장 첨부 및 별첨 서류 목록 ····· 214P

Ⅲ. 제2세부연구개발과제 연구결과
제1장 제2세부연구개발과제의 최종 연구개발 목표 ····· 219P
제2장 제2세부연구개발과제의 연구개발 내용 및 방법 ····· 228P
제3장 제2세부연구개발과제의 최종 연구개발 결과 ····· 232P
제4장 제2세부연구개발과제의 연구결과 고찰 및 결론 ····· 302P
제5장 제2세부연구개발과제의 연구성과 ····· 304P
제6장 기타 중요변경사항 ····· 306P
제7장 참고문헌 ····· 307P
제8장 첨부 서류 목록 ····· 308P

Ⅳ. 첨부 서류

연구결과보고서 요약문

연구과제명	데이터세트 유형 전자기록의 장기보존기술 연구		
중심단어	데이터세트, 장기보존, 마이그레이션, 에뮬레이션		
주관연구기관	전북대학교 산학협력단	주관연구책임자	양 동 민
연구기간	2019. 05. 02 - 2019 . 11. 29.		

□ 연구목표

세부구분	연구목표
제1세부	· 데이터세트 유형 전자기록 현황 및 장기보존기술 조사 · 데이터세트 보존포맷 선정체계 수립 및 보존포맷 선정 · 국산 DBMS 큐브리드 대상 보존포맷 변환기능 개발 및 변환·복원 검증 · 테스트베드 구축·시험 결과에 따른 보존적합방식 제안
제2세부	· 데이터세트 유형 전자기록 현황 및 장기보존 기술 조사 및 분석 · 보존방식에 따른 기술적합도 검증 테스트베드 구축 및 데이터세트 에뮬레이션 시험

□ 각 세부연구과제별 연구 결과

ㅇ 제 1세부연구과제

- (조사팀M) 데이터세트 유형 전자기록 현황 및 장기보존기술 조사 및 분석
  * SIARD 2.1 표준 및 SIARD Suite(오픈소스 프로젝트) 라이선스 정책 조사 및 분석
  * SIARD 해외사례 및 공공기관 행정정보시스템 형태·운영·관리 현황 조사 및 분석
- (기준팀) 데이터세트 보존포맷 선정체계 수립 및 보존포맷 선정
  * 전자기록 보존포맷 선정을 위한 공통기준
  * RDB형 및 Non-DB형 데이터세트 보존포맷 선정을 위한 고유기준
- (개발팀) 국산 DBMS 큐브리드 대상 보존포맷 변환 · 복원 기능 개발
  * 큐브리드 확장 SIARD Suite 개발, 빌드 및 실행 매뉴얼 작성 완료
- (검증팀) 데이터세트 유형 전자기록 보존포맷 변환·복원 검증
  * 보존포맷 변환·복원 검증을 위한 “사전”, “검증”, “DB크기 검증“ 및 ”실데이터 검증“ 시험
- (총괄팀) 테스트베드 구축·시험 결과에 따른 보존적합방식 제안
  * SIARD 장단점 분석, SIARD의 보존포맷 활용을 위한 오픈소스 수정·보완 사항 도출
  * SIARD기반으로 데이터세트 보존 방안 제안

ㅇ 제 2세부연구과제

- (조사팀E) 데이터세트 유형 전자기록 현황 및 장기보존 기술 조사 및 분석
  * 디지털 양피지, 올리브 프로젝트, 디지털 포렌식에서의 에뮬레이션 사례 조사
  * UNIX 계열 가상화 기술과 대안, Unix To Linux 전환 방법, 고려사항, 전환 사례 조사
- (구축팀) 기술적합도 검증을 위한 테스트베드 구축
  * 오픈소스 활용을 위해 오픈스택(Openstack) 기반 클라우드 및 에뮬레이션 시험 환경 구축
- (시험팀) 데이터세트 유형 전자기록의 에뮬레이션 시험
  * 에뮬레이션 시험 대상 시스템 선정, 에뮬레이션 환경으로 전환 시험 및 검증
  * 에뮬레이션 절차 및 정합성 검증 항목 도출

Summary

Title of Project	Study on long-term preservation technology of dataset-type electronic records		
Key Words	Dataset, Long-term Preservation, Migration, Emulation		
Institute	Jeonbuk National University	Project Leader	Yang, Dongmin
Project Period	2019. 05. 02 - 2019 . 11. 29.		

☐ Research Purpose

Tast Number	Research Purpose
Research Task Number 1	· Focused on migration, survey and analysis of dataset-type electronic record status and long-term preservation technology · Establishment of selection system for dataset preservation format and selection dataset preservation format · Development of conversion and restoration function for domestic DBMS CUBRID · Proposal of preservation method according to testbed construction and test result
Research Task Number 2	· Focused on emulation, survey and analysis of dataset type electronic records and long-term preservation technology · Establishment of testbed for verification of technical consistency and experiment of dataset emulation test according to the preservation method

☐ Research Result

° Research Task Number 1

- (Survey Team M) Focused on migration, survey and analysis of dataset-type electronic record status and long-term preservation technology
  * Survey and analysis of SIARD 2.1 standard and open source project (SIARD Suite) license policy
  * Survey and analysis of SIARD overseas cases and system/operation/management of public institution administrative information system

- (Criteria Team) Establishment of selection system for dataset preservation format and

selection dataset preservation format

* Common criteria for Selecting Electronic Record Preservation Format
* Unique criteria for selecting dataset preservation formats for RDB and non-DB

- (Development Team) Development of conversion and restoration function for domestic DBMS CUBRID
  * Development of SIARD Suite for CUBRID, and creation of building and execution manual

(Verification Team) Verification of conversion and restoration for dataset-type electronic record preservation format

* Conduction of 'Pre-test', 'Verification test', 'DB Size verification test' and 'Real Data verification test' for verification of conversion and restoration for preservation format

- (Integration Team) Proposal of preservation method according to testbed construction and test result
  * Analysis of SIARD advantages and disadvantages
  * Derivation of modifications of open source code to utilize SIARD's preservation format
  * Proposal of dataset preservation plan based on SIARD

° Research Task Number 2

- (Survey Team E) Focused on emulation, survey and analysis of dataset type electronic records and long-term preservation technology
  * Survey of emulation cases on digital Vellum, olive projects, and digital forensics
  * Investigation of UNIX-based virtualization technologies and alternatives, Unix To Linux transition methods/considerations, and transition cases

- (Implementation Team) Establishment of testbed for verification of technical consistency
  * Building an Openstack-based cloud and emulation test environment to utilize open source

- (Experiment Team) Experiment of dataset emulation test
  * Section of emulation test system selection,
  * Experiment and verification of system transition into an emulation environment
  * Derivation of emulation procedures and consistency verification items

총괄연구개발과제 연구결과

데이터세트 유형 전자기록의 장기보존기술 연구

양 동 민

전북대학교 산학협력단

제1장 총괄연구개발과제의 최종 연구개발 목표

1. 총괄연구개발과제의 목표

가. 연구배경 및 필요성

○ **(행정정보데이터세트에 대한 기록관리 필요)** 철건 구조의 표준전자문서 중심으로 최적으로 설계된 기록관리시스템의 행정정보데이터세트까지로 기록관리 범위 확장 검토 필요

- 국가기록원의 기록관리시스템들은 표준전자문서를 기반으로 수행되는 중앙정부 및 지방부처의 행정업무에 최적화되어 있음

- 철건 구조로 이루어져 있는 표준전자문서에 최적화되어 있기 때문에 다른 전자 파일(데이터세트, 시청각기록물 등)에 대한 기록관리의 확장성을 위한 연구 필요하며 특히, 실시간으로 엄청나게 생산되는 행정정보데이터세트에 대한 방안이 가장 시급

○ **(데이터세트에 최적화된 구체적인 마이그레이션 전략 및 기술 검증 필요)** 데이터세트에 적합한 문서보존포맷 선정 및 실증적인 실험 및 검토 필요

- 다양한 유형의 DBMS의 데이터세트를 수용하기 위해서는 정량적인 평가가 가능한 데이터세트 문서보존포맷 선정기준체계의 정립 필요하며, 데이터세트 문서보존포맷의 선정기준 및 프로세스 등 선정체계에 대한 구체화가 필요

- 선정기준으로 도출된 데이터세트의 문서보존포맷에 대한 객관적이고 실증적인 확인 필요하며, 실제 테스트베드에서 다양한 DBMS에 대해서 선정된 데이터세트 문서보존포맷의 유형, 규모, 환경 등에 대한 검증 작업이 필요

○ **(데이터세트 유형의 전자기록 관리를 위해 에뮬레이션 전략으로의 확장 필요)** 최고 수준의 무결성.진본성을 보장하기 위한 테스트베드 기반 연구가 필요

- 에뮬레이션 방식은 생산된 당시의 형태와 기능 그대로를 재현할 수 있는 가장 좋은 방법이지만 고도의 IT기술이 요구되고 비용도 많이 소요된다는 단점을 지님

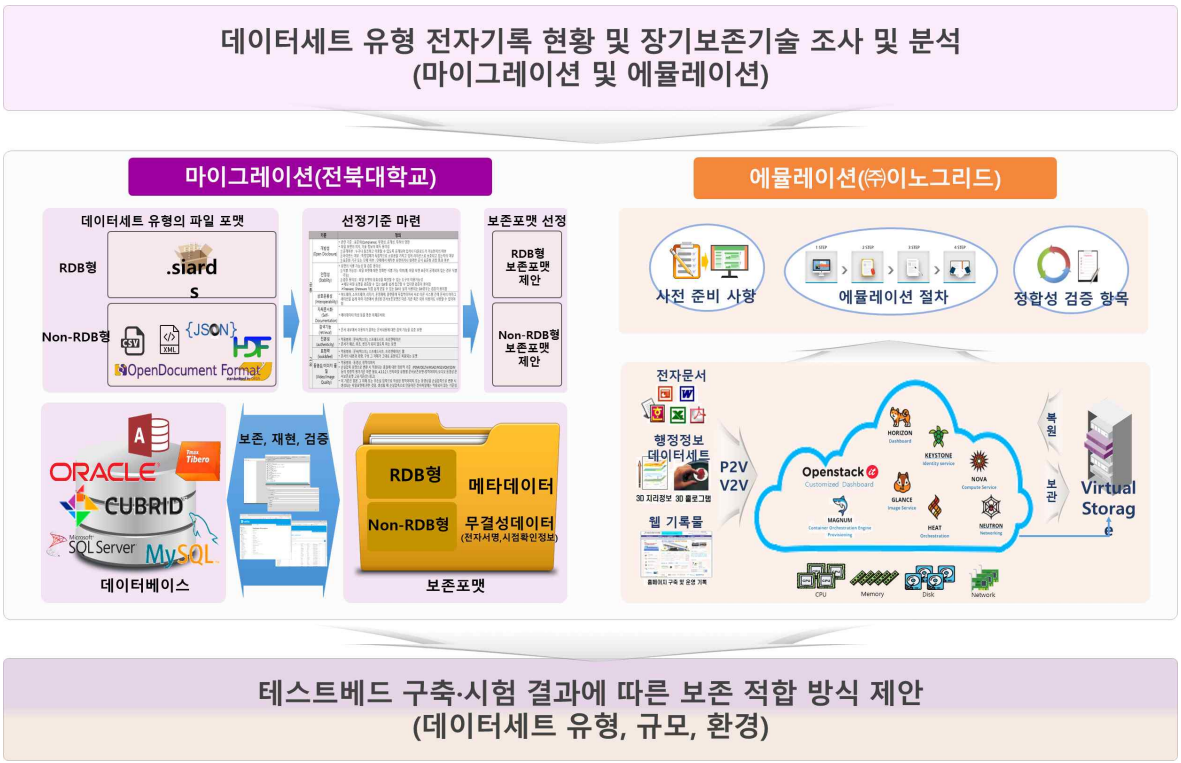
- 최근 클라우드 기반 가상화 기술의 발달로 비용이 많이 절감되어 산업계에서는 상용화가 이뤄져 널리 사용되고 있으며 Amazon의 AWS(Amazon Web Service)가 대표적임

- DBMS의 데이터세트는 독자적으로 사용되지 않고 WAS/WEB서버들과 연계되어 운용되는 경우가 많기 때문에 데이터세트만 보존하기보다 시스템 전체를 보관하기 위한 에뮬레이션 방식이 적합한 경우가 많음

- 클라우드, 모바일 등 컴퓨팅 환경의 변화를 고려하여 동적 요소를 포함한 전자문서의 영구보존 및 에뮬레이션 실증을 위해 테스트베드 구축과 함께 데이터세트의 유형, 규모, 환경 등에 따른 실험과 검토가 필요

나. 연구목표

구분	내용
최종목표	· **데이터세트 유형 전자기록의 장기보존**을 위해 **마이그레이션**과 **에뮬레이션 테스트베드**를 구축하고 데이터세트 **유형, 규모, 환경** 등에 따라 **실험을 통해 검증**하여 **최적의 보존방식**을 도출하는 것을 목표로 함
세부목표	**주요기능** · 데이터세트 유형 전자기록 현황 및 장기보존기술 조사 및 분석 · 데이터세트 유형 전자기록 보존포맷 선정기준 및 선정 · 다양한 DBMS 대상으로 데이터세트 유형 전자기록 보존포맷 변환 검증 시험 및 국산 DBMS 대상변환 기능 개발 · 보존방식에 따른 기술적합도 검증 테스트베드 구축을 통한 데이터세트 유형 전자기록의 에뮬레이션 시험 · 테스트베드 구축·시험 결과에 따른 보존적합방식 제안



<그림 1> 최종 연구 목표

1.2 총괄연구개발과제의 목표달성도

1.2.1 제1세부연구과제: 데이터세트 유형 전자기록 보존포맷 선정 및 테스트베드 구축・시험・검증

연구개발 추진내용			연구개발 일정							달성도
			5	6	7	8	9	10	11	
제1세부과제	마이그레이션	(조사팀M) 데이터세트유형 전자기록현황 및 장기보존기술 조사								100%
		· 데이터세트전자기록마이그레이션 장기보존기술								100%
		· 공공기관 행정정보시스템 형태・운영・관리현황								100%
		(기준팀) 데이터세트유형 전자기록 보존포맷 선정기준, 평가체계 수립 및 보존포맷 선정								100%
		· 데이터세트유형 전자기록특성 도출 및 보존포맷 선정기준 수립								100%
		· 보존포맷 선정기준에 따른 평가체계 개발 및 보존포맷제시								100%
		(검증팀) 데이터세트전자기록보존포맷변환 검증 시험 (총 4종)								100%
		· 다양한 DBMS 테스트베드 구축								100%
		· 테스트베드 기반 보존포맷변환 검증 시험								100%
		(개발팀) 국산 DBMS 대상 변환 기능 개발								100%
		· 오픈소스 기반 국산 DBMS의 보존포맷변환 SW 개발								100%
		(총괄팀) 테스트베드 구축·시험 결과에 따른 보존적합방식 제안								100%
		· RDB와 Non-RDB에 따른 보존포맷제안								100%
		· 데이터세트유형 전자기록보존·활용을 위한 지침								100%
		(총괄팀) 사업관리								100%
		· 월간업무보고								100%
		· 보고서, 발표자료 등 작성								100%

1.2.2 제2세부연구과제: 클라우드 기반 전자기록의 장기보존기술개발 테스트베드 구축 및 에뮬레이션 시험·검증

연구개발 추진내용			연구개발 일정							달성도
			5	6	7	8	9	10	11	
제2세부과제	에뮬레이션	(조사팀E) 데이터세트 유형 전자기록 현황 및 장기보존 기술 조사 및 분석(에뮬레이션 중심)								100%
		· 에뮬레이션 사례 조사								100%
		· UNIX 관련 에뮬레이션 사례 및 방안 조사								100%
		(구축팀) 기술적합도 검증을 위한 테스트베드 구축								100%
		· 하드웨어 시스템 도입 및 구성								100%
		· 클라우드 환경 구축								100%
		· 에뮬레이션 시험 환경 구축								100%
		(시험팀) 데이터세트 유형 전자기록의 에뮬레이션 시험								100%
		· 에뮬레이션 시험 대상 시스템 선정								100%
		· 선정된 시스템 별 에뮬레이션 시험 검증								100%
		· 에뮬레이션 절차, 정합성 검증 항목 도출								100%
		· 전자기록 보존 및 활용을 위한 지침 작성 지원								100%
		사업관리								100%
		· 주간 · 월간업무보고								100%
		· 최종보고서작성								100%

1.3 국내・외 기술개발 현황

1.3.1 국외연구동향

가. 장기보존 정책 및 전략

○ 호주의 NAA(National Archives of Australia)는 2018년 1월에 디지털 보존정책(Digital Preservation Policy)을 발표

- NAA는 일반적인 문서 및 이미지뿐만 아니라 새로운 포맷인 전자메일, 시청각 기록, 혼합된 미디어(웹사이트 등), 구조화된 데이터 세트 등 다양한 전자기록물 유형을 선호·허용·위험 포맷으로 선정하여 보존

- NAA는 기본적으로 마이그레이션 전략을 사용하지만, 원본 비트스트림을 보존하고 모든 버전을 보유하여 지속적인 접근성을 보장

- NAA은 장기보존을 위해서 Manifest Maker, XENA(Xml Electronic Normalising for Archives), DPR(Digital Preservation Recorder), Checksum Checker 개발

- NAA는 Manifest Maker를 통해 체크섬을 생성하고, Checksum Checker를 통해 검증하며, DPR을 사용하여 기록의 바이러스, 무결성을 체크하고 보존포맷(XENA)으로 변환(NAA, 2011)

○ 캐나다의 LAC(Library and Archives Canada)는 2017년 11월 디지털 보존프로그램을 위한 전략(Strategy for Digital Preservation Program) 개발

- LAC가 개발한 디지털 보존프로그램 전략은 국제 표준인 ISO 14721:2012 OAIS 참조 모형을 통하여 디지털 아카이브의 기능과 역할을 설명하고 디지털 보존프로그램을 구현하는데 필요한 핵심 요소 정의

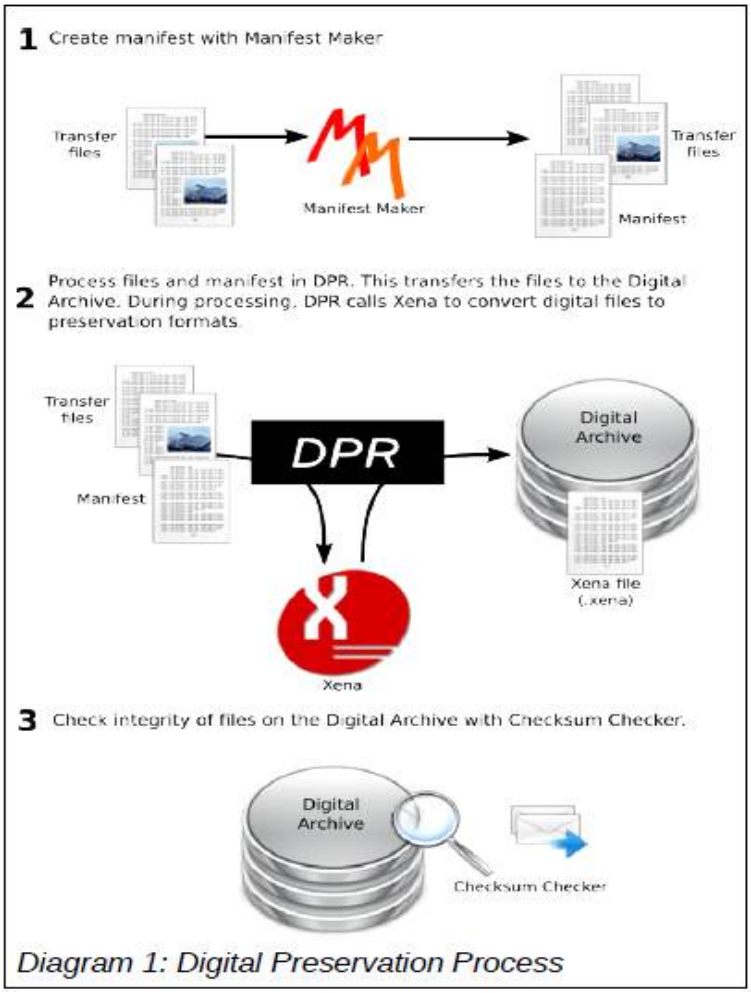
- LAC의 장기보존 정책은 디지털 보존프로그램을 개발하기 위한 로드맵을 설계에 중점

- LAC는 마이그레이션 전략 채택하고 보존포맷 요건을 다음과 같이 정의(LAC, 2017)

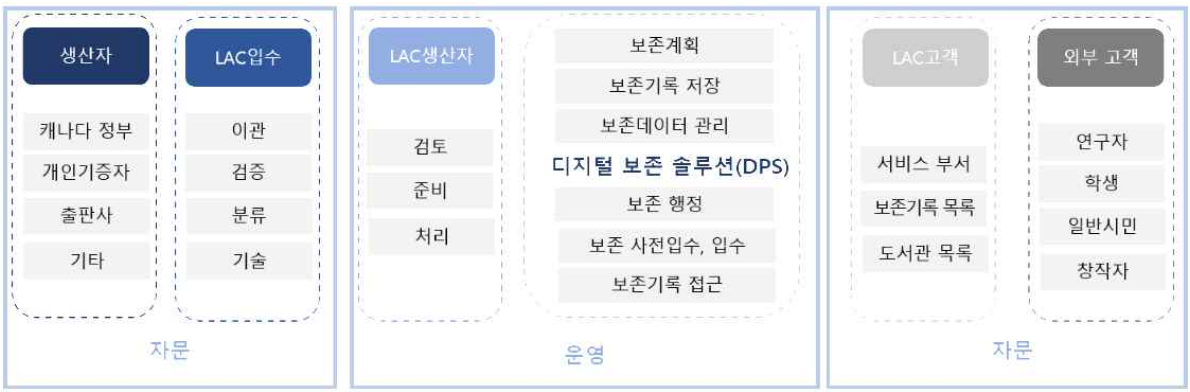
요건	내용
공개성/투명성	· 파일포맷 지식과 기술정보 축적이 용이
채택	· 국가도서관, 아카이브 등에서 공식으로 채택
안정성/호환성	· 이전/이후 기종과 호환, 파일 변형으로부터 보호되는 정도 · 시간 경과에 따른 포맷 업데이트 또는 대체 버전의 상대적 빈도
의존성/상호운용성	· 포맷이 특정한 하드웨어 또는 소프트웨어에 의존하는 정도

<표 1> LAC의 보존포맷 요건

- LAC는 포맷 마이그레이션 정책을 통해 구형화 위험 포맷 및 매체에 대한 선호 포맷 가이드라인 제시



<그림 2> Digital Preservation Process (NAA, 2011)



<그림 3> LAC의 디지털 보존프로그램 (LAC, 2017)

○ 2011년 영국 TNA(The National Archives) '디지털 보존 정책아카이브즈를 위한 지침'(Digital Preservation Policies: Guidance for archives)'

- 2017년 '디지털 전략 2017-2019'(Digital Strategy2017-2019)에서 보완
- TNA는 현재 문서, 이미지, 이메일, 비디오, 웹사이트 등에 집중하고 있으나 구조화된 데이터세트와 컴퓨터 코드까지 보존할 수 있도록 확대하고 이를 위한 기술 개발 고려
- TNA는 장기보존 전략으로서 원본 비트를 유지하는 에뮬레이션 전략 선호
- TNA는 보존도구로서 DROID, PRONOM, COPTR 개발

기준	개념	특징	환경
DROID	포맷 식별 도구	· 내부 서명을 이용하여 식별 · 오픈소스, 전 세계적으로 널리 사용 · 현재 6.4v · 버전, 나이 및 크기, 마지막으로 변경된 시기를 알 수 있음 · csv파일로 내보내기 가능	JAVA 1.7 또는 1.8 Standard Edition (SE) 환경
PRONOM	포맷 정보 레지스트리	· 정부 부서에 레코드를 저장할 때 사용되는 파일 포맷 정보를 관리 · 1,300개 이상 개별 파일 형식 항목을 포함 · 현재 PRONOM 6.2v · 마이그레이션 전략을 시행 시 중요한 정보 제공	Window 2000
COPTR	보존 도구 레지스트리	· 보존 전문가가 특정 보존 작업을 수행하는 데 필요한 도구를 찾고, 평가하는 도구 · 현재 445가지 도구 보유 · OAIS 참조모델과 DCC의 디지털 큐레이션 생애주기 모델 참조	-

<표 2> TNA 디지털 보존 도구

○ 미국 NARA(National Archives)은 2017년 8월 디지털기록 보존전략(Strategy for Preserving Digital Archival Materials)을 발표함

- 디지털기록 보존전략에서 전략으로써 6가지 전략 제시

(1) 표준과 절차의 문서화(Documentation of Standards and Procedures)

(2) 우선순위(Prioritization)

(3) 파일 관리(File Management)

(4) 진본성(Autnenticity)

(5) 보존 메타데이터(Preservation Metadata)

(6) 조직적인 관계(Organizational Relationships)

- NARA는 보존활동을 디지털 보존 인프라 기반 구조, 데이터 무결성, 포맷 및 매체 지속성, 정보보안으로 구분하여 제시

- NARA는 다양한 유형의 전자기록물을 보존하기 위해 마이그레이션 전략을 채택하고 포맷의 경우 선호, 허용, 위험 포맷으로 구분하여 체계적으로 관리

○ 스위스의 SFA(Swiss Federal Archives)는 2009년에 발표된 'digital archiving policy'와 2015년에 발표된 'Federal Archives Strategy 2016 – 2020'에서 나타남

- SFA는 텍스트 문서에서 사진, 녹음, 매우 복잡한 데이터베이스에 이르기까지 다양한 형태를 보존

- SFA는 보존 전략으로 <표 3>의 3가지 전략을 제시

전략	내용
마이그레이션 원칙	· SFA는 마이그레이션 전략을 사용하여 전자기록물을 필요에 따라 변화하는 환경에 맞추어 선정된 보존포맷으로 변환하여 보존 · 무손실 변환을 강조하며, 모든 변환은 변경 사항이 기록되며, 문서는 어떤 경우에도 복원 가능하도록 함 · 에뮬레이션 전략을 사용하지 않고, 원래의 하드웨어와 소프트웨어를 보존하지 않음
어플리케이션 분리	· SFA는 특정 IT환경(응용프로그램, 데이터베이스, 운영체제, 하드웨어)에서 데이터를 분리하는 전략 추구 · 응용프로그램은 보존하지 않음 · 특정 응용프로그램(예. 데이터베이스)은 기록관리 업무 시 검증하고 데이터를 사용할 필요가 있을 때 보존 할 수 있음(예: 데이터모델)
원본 데이터의 매체는 저장하지 않음	· 디지털 문서는 원본 데이터 매체를 고려하지 않고 보관 · 보존 프로세스가 완료될 시 원본 데이터 매체에서 데이터를 삭제하고, 이관이 완료되면 이관에 사용된 보존 매체는 폐기되거나 반환됨

<표 3> SFA의 3가지 전략

나. 문서보존포맷 기술 현황

○ 개념

- 문서가 생산된 당시의 어플리케이션이 없어도 해당 문서의 내용과 외형을 그대로 재현하여 내용보기를 가능하게 하는 포맷(예 : PDF/A, SIARD 등)

○ SIARD 2.0 (Software Independent Archiving of Relational Databases)

- 유럽 E-ARK 프로젝트와 스위스 아카이브가 개발한 관계형 데이터베이스의 장기보존포맷 표준

- 특징

· 2013년에 eCH-0165 표준으로 공인되었고, SIARD 2.0은 SIARD 1.0을 기반으로 하여 개발되었기 때문에 SIARD 1.0과 호환성을 가지고 있음

· 모든 SQL:2008 타입 지원, 특히 사용자 정의 데이터 타입도 지원

· 정규 표현식을 사용한 데이터 타입 정의를 위한 보다 명확한 검증 규칙

· “file:” URI를 사용하여 SIARD 파일 외부에 있는 대형 객체 저장 기능 지원

· “deflate” 방식의 압축 기법 지원

- SIARD Suite 지원 DBMS

· MS Access 2007 or higher

· DB/2 8 or higher

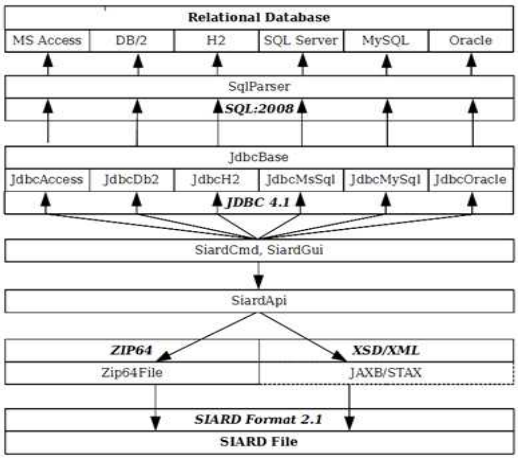
· H2 database 1.4 or higher

· MySQL (or MariaDB) 5.5 or higher

· Oracle 10 or higher

· SQL Server 2012 or higher

· Postgres or higher(2019년 6월 14일 추가)



<그림 4> SIARD Suite 구조

○ 문서보존포맷 기준 현황

- 해외 각국에서는 다양한 전자기록물 수용을 위해 여러 문서보존포맷 선정을 위한 문서보

존포맷 기준을 제시하고 있음

- 문서보존포맷 기준

· 개방성/투명성 : 파일 포맷의 지식, 기술 정보의 획득 용이성

· 채택 : 국제적으로 국립 도서관, 기록관 및 기타 기록유산기관이 공식적으로 채택된 정도

· 안정성/호환성 : 포맷의 버전 공개빈도 주기, 전후 버전의 호환가능성 및 변경·변질에 대한 탄력성 정도

· 종속성/상호운용성 : 포맷이 특정 하드웨어, 소프트웨어, 리더기에 의존하는 정도

· 표준화(규격화) : 포맷이 엄격한 공식 표준화 과정을 통과하는 정도

· 압축 : 압축 및 무손실 압축

· 특허의 영향 : 한 포맷으로 콘텐츠를 유지하는 아카이브 기관의 능력이 특허에 의해 금지 될 것이라 여기는 정도

· 기술 보호 메커니즘 : 신뢰할 수 있는 저장소에 의한 콘텐츠 보존을 방해하는 암호화와 같은 메커니즘 구현

· 메타데이터 지원 : 메타데이터 작성 등을 통한 자체문서화(self-documentation)

· 진본성 : 문서가 훼손, 위조, 변조가 되지 않도록 하는 포맷

· 표현력 : 문서의 내용과 외형, 구성 그 자체가 그대로 표현되고 복원되는 포맷

· 검색기능 : 문서 내부에서 이용자가 원하는 문서내용에 대한 검색 기능을 갖춘 포맷

기준	캐나다 (LAC)	INTERPARES 2 Project	미국 (LOC)	영국 (TNA)	미국 (NARA)	한국 (NAK)
개방성/투명성	○	○	○	○	○	○
채택	○	○	○	○	○	○
안정성/호환성	○			○		○
종속성/상호운용성	○	○	○	○		○
표준화(규격화)	○					
압축		○			○	
특허의 영향			○	○		
기술 보호 메커니즘			○		○	
메타데이터 지원			○	○	○	○
진본성						○
표현력						○
검색기능						○

<표 4> 해외 문서보존포맷 기준 현황

○ 전자기록유형별 문서보존포맷 선정 현황

- 해외 아카이브 포맷은 일반적으로 선호포맷과 허용포맷을 제시하고 있으며, 기록유형별로 다양한 보존포맷을 허용하고 있음

구분	선호포맷	허용포맷
문서	PDF/A-1, PDF/A-2, TXT(ASCII, UTF-8, UTF-16), BITS, EPUB, PDF/UA, PDF/A, PDF, ODF, ODT 등	PDF, DOCX, DOC, EPUB(2.0.1), XHTML or HTML with DOCTYPE, ODF, OOXML, RTF, Plain Text, 등
프레젠테이션	SVG, TIFF, ODP, PDF/A-1, PDF/A-2	PPT, PPTX, PDF/A-2
데이터세트	JSON, TSV, CSV, SQLITE, DB, DB3, ODS, ASCII text, XML	CDF, HDF, XLS, XLSX, ODS, EBCDIC, DBF, JSON, CSV, ASCII, UTF-8, UTF-16, SIARD(2.0)
정적이미지	SVG, TIFF, JP2(JPEG2000), DNG, BMP, GIF, JPG(JPEG/JFIF), PNG, JP2(Part1), PDF/A(PDF/A-1), PDF/A-2	PSD or PSB, NEF or CRW, JPF, EPS, DNG, GIF, JFIF with JPEG compression, PNG, JP2(JP2-part1), PDF/A-2, DICOM, ODG, Exif
오디오	WAVE, WAV, DSD, BWF, FLAC	WAVE, AIFF, MP3, WAV, AAC, FLAC, MPEG-1 layer3
동영상	DPX, AVI, MXF, DCDM, MOV	AVI, WMV, MPEG-4, MPEG-2, MXF, MOV, DCP, DCDM
CAD	X3D, STEP, AutoDesk's Drawing File, DXF	PDF/E, U3D, PRC, STEP, PDF/E, DXF, DWG
지리공간	ArcGIS, GML, GDAL, OGR, GTIFF, SHP&DBF, KML or KM2, GeoTiff, BIL, Band Interleaved by Pixel, BSQ, DEM, ESRI Arc/Info ASCII Grid, ESRI SHP	SHP, Ventor Product Format, E00, TerraGo Geospatial PDF, SHX, DBF, CCOGIF, DIG3, Geospatial PDF, IHO, GerraGoGeoPDF, GeoTIFF
웹	WARC, ARC, XHTML	ARC-1A, HTML
이메일	EML, MBOX, PST	XML, MSG, PST, EML, MBOX
캡슐화		ZIP, TAR, BagIt

<표 5> 국외 문서보존포맷 현황 - 선호포맷 vs 허용포맷

구분	미국NARA	캐나다 LAC	영국TNA	호주NAA	스위스SFA	중국SAC
문서(텍스트)	PDF, PDF/A-1, PDF/A-2, DOCX, ASCII, Unicode, OOXML, DOC,	EPUB, PDF, PDF/A-1, PDF/A-2, DOCX, ASCII, Unicode, ODF, DOC	PDF, DOCX, ASCII, Unicode, OOXML, DOC, DOC, ODT, OTT, SXW, RTF, WPD, WP6, WP, WP62, WP61, XSD, DOT	PDF, PDF/A-1, PDF/A-2, DOCX, ASCII, Unicode, DOC, ODT	PDF/A-1, PDF/A-2, ASCII, Unicode, XML	PDF, PDF/A, DOCX, RTF, WPS, JPG, TIF, PNG,
프리젠테이션	ODP, PDF/A-1, PDF/A-2, PPT, PPTX	ODP, PDF/A-1, PPT, PPTX	ODP, PPT, PPTX, OTP, SXI, SHW	ODP, PDF/A-1, PDF/A-2, PPT, PPTX		
데이터세트	JSON, CSV, XLS, XLSX, ODS, ASCII, XML	CSV, XLS, ODS, ASCII, EBCDIC, DBF, OTS, SXC, WKS, XLW, XLSB, XLSW	CSV, XLS, ODS	SON, CSV, XLS, XLSX, ODS, ASCII, Unicode, XML, EBCDIC, DBF, SIARD, MS Access	CSV	XLS, XML, DBF, SIARD, ET
정적이미지	TIFF, JP2, DNG, GIF, JPG, PNG, PDF/A, PDF/A-1, PDF/A-2, ODG, SVG, NEF or CRW, JPF	TIFF, JP2, DNG, GIF, JPG, PNG, PDF/A, DICOM	TIFF, JP2, DNG, BMP, GIF, JPG, PNG, ODG, SXD, CDR, VSD	TIFF, JP2, BMP, GIF, JPG, PNG, PDF/A-1, PDF/A-2, ODG, SVG, Exif	TIFF, JP2	TIFF, JPG, EPS, SVG, SWF, WMF, EMF, DXF, STEP
오디오	WAV/WAVE, BWF, FLAC, AIFF, MP3	WAV/WAVE, BWF, AIFF, MP3, AAC, OGG(OGA), WMA, MID, MP4	WAV/WAVE, MP3, OGG(OGA), WMA(ASF), MID, MP4	WAV/WAVE, BWF, FLAC, AIFF, MPEG-1 layer3, MPEG-2 layer3	WAV/WAVE	WAV/WAVE, MP3
동영상	DPX, AVI, WMV, MPEG-4, MPEG-2, MXF,	DPX, AVI, WMV, MPEG-4, MPEG-2, MXF, DCDM, MOV, DCP	WMV, MPG, MXF, MOV, MP4, WEBM, SWF, M1V, M2V, OGG, JPEG2000	DPX, AVI, WMV, MPEG-4, MPEG-2, MXF, DCDM, MOV	MPEG-4, FFV1	AVI, MPG, MXF MP,
웹	WARC, ARC			WARC, ARC, HTML, XHTML	PDF/A, SIARD, XML	HTML
이메일	EML, MBOX, PST, MSG, XML	EML, MBOX, PST, MSG	EML, PST, MSG	EML, MBOX, PST, MSG		EML
CAD	STEP, PDF/E, X3D, U3D, PRC	STEP, PDF/E, DXF		STEP, PDF/E, DXF, DWG		
지리공간	GML, GTIFF, KML, ESRI ArcInfo Export (E00), TerraGo Geospatial PDF, ESRI SHP(ESRI Shapefile)	GML, GTIFF, KML, ESRI ArcInfo Export (E00), TerraGo Geospatial PDF, ESRI SHP(ESRI Shapefile)		GML, GTIFF,		

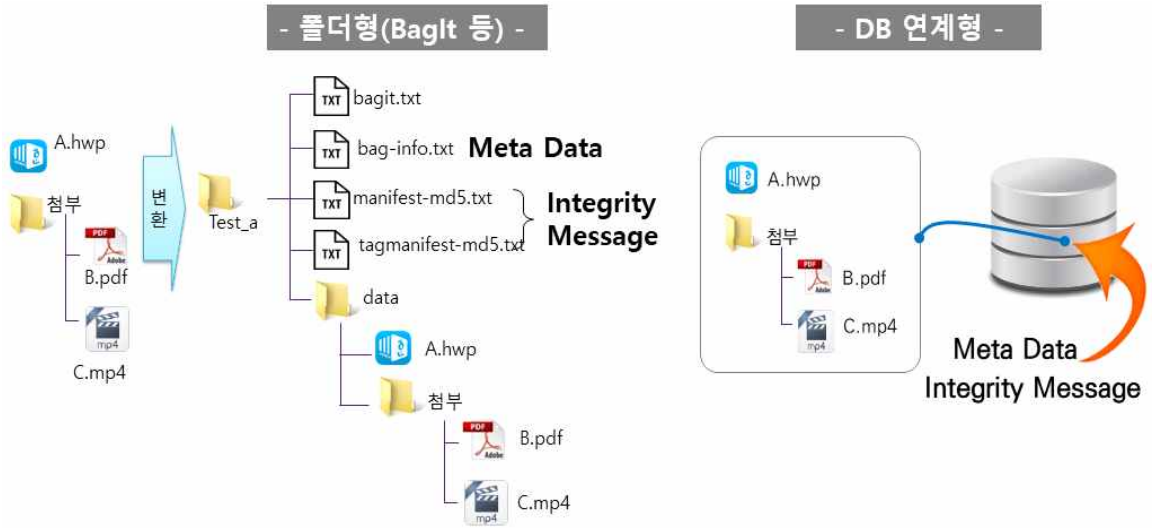
<표 6> 전자파일 유형별 문서보존포맷 국외 현황

다. 장기보존포맷 기술 현황

○ 개념

- 전자기록물의 진본성과 무결성을 보장하고 장기간 안전하게 보장하기 위해 전자기록물의 원문, 문서보존포맷, 메타데이터, 전자서명을 하나의 패키지로 구성한 포맷(예 : NEO, BagIT, Xena, AAP 등)

- 전자기록물의 데이터 무결성, 진본성, 신뢰성을 보장하기 위해 “메타데이터”와 “무결성 메시지”를 전자기록물(들)과 함께 하나의 “파일” 또는 “폴더” 로 묶어서 장기보존포맷으로 변환



<그림 5> 장기보존포맷 변환 유형

○ BagIT

- 미국 Library of Congress과 스탠포드 대학교에서 전자파일의 이관을 위해 개발된 포맷으로 전자파일의 종류와 포맷 등에 종속적이지 않음

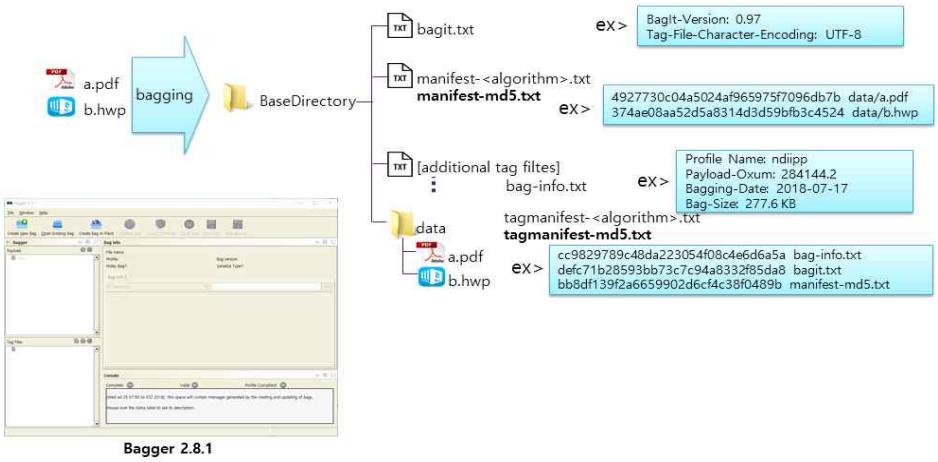
- 특징

· 하나의 폴더(Bag)로 묶는 패키지 방식

· Payload(digital files) + Tags(metadata)

· IETF Draft (https://datatracker.ietf.org/doc/draft-kunze-bagit/)

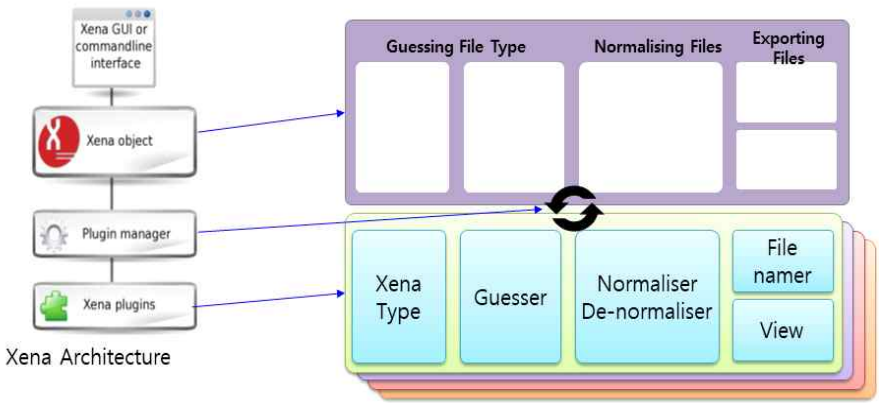
· Checksum 방식(MD5, SHA-1, SHA-256, SHA-512)



<그림 6> BagIT 구조

○ Xena(XML Electronic Normalising for Archives)

- 호주 국가기록원에서 개발한 디지털 보존 소프트웨어로 호주 NAA의 아카이빙 시스템인 DPSP(Digital Preservation Software Platform)의 구성요소 중 하나
- 하나의 파일을 XML 파일 안에 포함시키는 패키징 방식 이며, 패키징하는 기능을 JAVA API 로 제공
- Guesser가 파일 포맷 인식, Normaliser가 XML 기반 Xena 포맷으로 Base64로 인코딩
- 주기능: 개방형 파일 포맷에 기반한 적절한 보존 파일 포맷 형태로 변환
- Xena에서는 Plain text 카테고리에 스타일시트, 데이터베이스, SQL 등을 포함하고 있으며 파일 보존 시 개방형 파일 포맷으로 Unicode 또는 ASCII 코드로의 변환을 추진
- Payload(digital files) + Tags(metadata), Checksum(SHA-512)



<그림 7> Xena 구조

○ AAP(Archival Asset Package)

- 미국 국립기록관청(NARA)의 전자기록물 아카이브인 ERA(Electronic Records Archives)의 장기보존 패키지
- 장기보존 메타데이터(ACE: Asset Catalog Entry)와 디지털 콘텐츠를 인캡슐레이션한 형태
- 특징
  · 장기보존 메타데이터와 디지털 콘텐츠를 인캡슐레이션한 형태
  · 파일 인캡슐레이션을 위해 ZIP과 URL 옵션 사용 가능
  · 핸들(Handle), DOI, PURL 등 현재 및 미래의 표준 프로토콜과 통합 가능
  · 상이한 레벨의 ACE 구조로의 접근 제공

기관명	장기보존포맷	구성요소	플랫폼	유형
영국 TNA	BagIt/Dspace	원문파일+보존파일+MD+Checksum	Preservica	폴더형
미국 NARA	AAP(자체패키지)	원문파일+보존파일+MD	ERA	-
스위스 SFA	BagIt/Dspace	원문파일+보존파일+MD+Checksum	Preservica	폴더형
호주 PROV	VEO	원문파일+보존파일+MD+전자서명(Base64 인코딩)	PROVDigitalArchiveSystem	파일형
호주 NAA	XENA	원문파일+보존파일+MD+Checksum	DPSP	파일형
국가기록원	NEO	원문파일+보존파일+MD+전자서명(Base64 인코딩)	CAMS	파일형
UNESCO	BagIt	원문파일+보존파일+MD	Archivematica	폴더형
캐나다 벤쿠버 시 아카이브	BagIt	원문파일+보존파일+MD	Archivematica	폴더형
미의회도서관	BagIt	원문파일+보존파일+MD+Checksum	Archivematica	폴더형
뉴욕현대미술관	BagIt	원문파일+보존파일+MD	Archivematica	폴더형

<표 7> 국외 장기보존포맷 현황

(출처 : 2018년 국가기록원 1차 연구세미나 재편집)

라. 보존프로젝트 현황

○ ERA 2.0(Electronic Records Archives)

- 전자기록에 대한 관리 및 보존을 위해 논의되고 있는 수많은 문제들을 해결하기 위하여 ERA 2.0이 추진됨

- ERA 2.0의 개발 범위에는 모든 핵심 디지털 보존 기능과 자동화된 보존조치 및 매체전환과 같은 대량 보존조치를 포함하는 기능을 크게 확장하기 위한 프레임워크, 레거시 시스템과 분류된 시스템 개발이 포함됨

- NARA가 다양한 전자기록을 처리하고 보존하는 데 필요한 여러 접근 방식을 지원하기 위해 시스템의 유연성을 증가시켜야 한다는 필요성에 따라 스케줄링에서 공개 접근에 이르기까지 기록의 생성주기 효율성을 향상시키는 것을 목표로 다음 내용 수행

· 프로그램 비용을 절약 및 안정화

· 생산성 향상 및 지원 기능 강화

· 보존 및 접근을 위해 보관 프로세스와 워크플로우 전반에서 공동 작업을 향상시킴

· 디지털 객체의 안전성, 신뢰성 확보, 안전한 “장기적인” 보존을 수행함

· 보존된 디지털 객체를 검색, 확인할 수 있는 기능을 향상시킴

· 이관, 인수/거부, 법적 보관권(custody) 업데이트를 위한 워크플로우 효율화

· 디지털 객체의 수집, 저장 및 검색의 기존 한계점 극복

· 보존 활동의 증가하는 백로그에 대한 고려

- ERA 1.0과의 주요 차이점

· 모듈화 : 시스템의 전반적인 복잡성을 줄이고 업무처리 도구들의 유연성을 향상시키기 위하여 ERA를 현대화하는 모듈식 접근법을 채택하여 기관에 대한 자체 지침과 끊임없이 변화하는 연방 전자기록관리 요구사항을 지원

· 클라우드 도입 : NARA는 ERA 2.0 개발에 적극적으로 클라우드 기술을 활용하기로 결정

· 제한된 워크플로우에서 벗어남

- ERA 2.0의 모듈 : DPE(Digital Processing Environment), DOR(Digital Object Repository), BOM(Business Object Management)

· 각 모듈(DPE, DOR, BOM)은 독립적인 코드 기반이며 목표는 시스템의 다른 측면에서 종속 코드가 변경될 경우 최소로 업데이트되거나 대체될 수 있는 모듈을 구현하는 것임. 사용자 정의 JAVA 응용 프로그램이지만 스택의 오픈소스 도구를 활용함

모듈	내용
DPE	· DPE의 구성요소는 모든 종류의 디지털 자료를 업로드하고, 검증과 업무처리를 위한 다양한 소프트웨어 도구를 제공하며(필요에 따라 도구 교환가능), 메타데이터를 생산하고 편집하는 기능 제공 · 이용자가 업무처리된 디지털 자료의 패키지가 보존을 위한 DOR 구성요소로 제출하는 것을 허용하는 기능 지원 · DPE는 NARA의 아키비스트가 다양한 디지털 자료를 처리할 수 있도록 확장 가능한 소프트웨어 도구 카탈로그와 함께 유연한 환경을 제공해야 함

DOR	· DOR 구성요소는 DPE에서 처리된 디지털 자료를 받아들이고, 확장가능하고 안전한 기록 관리 저장소를 제공하며, 상세검색과 통합검색(discovery) 기능 제공 · NAC(National Archives Catalog)를 통한 공공접근과 보존을 위한 추가 업무 처리를 위해 다시 DPE에 디지털 자료를 제공하는 기능 지원 · 기록에 대한 공공접근은 National Archives Catalog를 통해 가능하지만 DOR은 스태프를 위한 메타데이터와 기록 내용 모두에 대한 고급 검색기능을 포함함 · 이는 개인 식별 정보(PII), 분류 상태 및 대응되는 기록을 찾기 위한 특별한 접근, FOIA(Freedom of Information Act) 요청, 소송을 지원하기 위해 법에 의한 접근 요청이나 아직 공개적으로 이용 가능하지 않은 정부 정보에 대한 접근 제공을 위해 매우 세밀한 검색 요구사항을 지원 · DOR은 기록의 고정, 객체 버전관리, 검색, 감사 및 보고를 포함한 모든 홀딩(holding) 관리 및 보존 기능을 제공함 · DOR은 데이터 웨어하우스에서 자동 또는 수동으로 수행되는 모든 작업에 대한 감사추적을 유지함
BOM	· BOM은 일정관리, 이관 그리고 보관 프로세스를 다루는 비즈니스 객체를 관리하기 위해 안전하고 유연한 응용 프로그램을 제공해야 함 · 일정관리와 NARA에 정부기록의 물리적 및 법적 양도를 위한 현재 온라인 양식과 승인 워크플로우는 ERA의 BOM 구성요소에 의해서 제공됨 · 현재 시스템에는 다른 기록 유형을 제외한 연방 기록에 대한 단일 워크플로우만 포함됨 · ERA 2.0은 연방, 대통령. 입법부 및 사법부 기록, 디지털화된 아날로그 기록 그리고 기증된 자료의 여러 워크플로우로 구성되며 요구사항의 변화에 따라 워크플로우를 쉽게 업데이트하거나 새로운 워크플로우를 인스턴스화할 수 있는 메커니즘을 제공함

<표 8> ERA 2.0 모듈

○ E-ARK(European Archival Records and Knowledge Preservation)

- 덴마크, 헝가리, 에스토니아, 노르웨이, 스웨덴, 포르투칼, 스페인 등 여러 국가가 참여하는 기록 관리 유럽 공동 프로젝트

- 목적 : OAIS 참조모델의 3단계인 입수(Ingest) 단계, 보존(Preservationa) 단계, 배포(Re-USe) 단계를 위한 플랫폼 제공

- 아카이빙 프로세스

· 기록 관리 시스템에서 보관해 온 기록물을 받음

· SIP Cration Tool 을 이용하여 E-ARK SIP 형태로 생성

· SIP-AIP 변환기에서 E-ARK AIP 형태로 변환되어 Digital Preservation System에 보관됨

· 접근 시 AIP-DIP 변환기에서 E-ARK DIP 형태로 변환되어 검색과 접근이 가능

- 메타데이터 : IP(Information Package)의 METS 사용

마. 행정정보데이터세트 관리 및 보존 전략 현황(국가기록원 , 2015)

○ 데이터세트 정의

- 영국

· 특정 목표를 위해 생성된 구조화된 데이터의 모음

· 다양한 포맷과 기술로 저장, 관리, 공표될 수 있음

· 데이터베이스, 스프레드시트, 텍스트파일, 웹사이트 또는 프린트된 종이

- 호주

· '데이터베이스로부터 받은 데이터를 주기적인 스냅샷 기법을 통해 이해하는 것'으로 호주 국가기록원의 특정 문서에서 언급

- 뉴질랜드

· 리스트, 테이블, 스프레드시트, 데이터베이스 등에 저장된 구조화, 암호화된 정보

· 알파뉴메릭 데이터, 공간데이터, 스펙트럴데이터, 통계 데이터, 구조화된 텍스트(서지 데이터, 데이터베이스 정보 등)

· 데이터세트 정의에서 언급된 데이터베이스는 데이터베이스 실제 콘텐츠, DBMS, 데이터베이스 애플리케이션을 포괄하는 의미임

○ 해외 다수의 국가에서는 데이터세트를 기록대상으로 선정하여 적합한 정책 및 기술을 통해 무결성, 진본성 보장 방안을 제시하고 있음

국가	내용
미국	· 연방정부에서 생산된 기록물을 NARA에서 관리하는 AAD(The Access to Archival Database)에 이관 - AAD : NARA에서 2003년 2월 12일부터 운영하고 있는 데이터베이스 아카이브 시스템으로 온라인을 통해 데이터 검색, 다운로드, 인쇄 가능 · 데이터세트는 Geospatial(특정 지역과 관련된 데이터)과 Non-geospatial 두 가지 유형의 데이터세트를 제공 · 메타데이터 : 더블린코어
독일	· CHRONOS : OAIS기법을 통해 데이터베이스를 아키이빙하기 위한 CSP의 상업적솔루션 · 데이터세트 정의는 따로 없으나, 데이터베이스만을 아카이빙 대상으로 함 · 아카이빙 프로세스 및 요구사항 - 시간을 기준으로 아카이빙을 원칙으로 함 - incremental archiving 방법을 사용 - 아카이빙 프로세스 : 습득(ingest), 데이터 관리(Data Management), 기록 저장(Archival Storage), 접근(Access), 보존계획(Preservation Planning), 관리(Administration) · 메타데이터는 Chronos 다큐멘테이션 및 홈페이지에서 administrative and technical

	metadata를 정의하고 있으나 세부적인 내용은 제공하지 않음 · 보존포맷 : text와 XML 파일로 구성된 ZIP 파일 구조 · 무결성, 진본성 보장 방안 - 해시코드를 사용하여 bit 단위로 보존포맷(AIP)에 대한 무결성 검증 - 타임스탬프를 이용한 무결성 검증
영국	· TNA는 NDAD(National Digital Archive of Dataset)을 통해 1997년부터 2010년까지 정부의 데이터세트를 아카이빙하고 있음 - NDAD 웹사이트를 통해 다운로드 하거나 또는 DVD로 구매를 위해 원본 포맷을 간단한 개방형 CSV 포맷으로 변환함 - NDAD 시스템이 2010년에 중지된 뒤 2010년 이후로 UK Government Web Archive가 쓰임 - NDAD는 데이터세트, 데이터세트의 맥락을 알아주고 해석에 도움을 주는 다큐멘테이션, 메타데이터를 보존 대상으로 획득 · 기록 대상 행정정보시스템 - 정부부처의 주요한 정책과 활동에 관한 시스템 - 정부의 의사 결정 프로세스 및 구조에 관한 시스템 - 국민 생활과 관련된 국가 정보 - 물리적 환경과 관련된 국가 정보 · 데이터세트 보존포맷 : 데이터베이스는 일반적으로 합의된 형식 (XML, CSV를 포함한 E-GIF 파일)으로 Export 할 수 있어야 함 · 데이터세트에 대한 평가 작업은 해당 주제나 분야와 관련된 선별정책을 통해 이루어짐
호주	· 소프트웨어를 이용해 디지털 기록을 보존하기 위해 DPSP(Digital Preservation Software Platform)를 개발하였으며, 디지털 기록에는 데이터베이스가 포함되어 있음 · 아카이빙 프로세스 및 요구사항 - 이관할 파일에 Manifest Maker를 이용하여 Manifest생성 - Xena를 호출해 보존포맷으로 변환하는 DPR Processing - Checksum Checker와 함께 무결성 확인과 데이터 손실을 모니터링 1) 문서이관 : 문서는 디지털 보관소에 이관되는 모든 기록물들의 Checksum과 Manifest를 포함 2) DPR processing : 기록물은 바이러스 및 무결성 체크를 통해 이상 유무를 파악하고, 어느 곳에 필요한지 확인 후에 Xena를 호출해 어느 보존포맷으로 변환할지 결정 3) 저장 : 기록물은 디지털 보관소에 저장되며 Checksum Checker와 함께 무결성 확인 및 데이터의 손실을 지속적으로 감시 · 메타데이터 : Xena 파일 메타데이터 사용, Wrapper schema, Package schema, NAA schema, Content-specific schema로 구성 · 보존포맷 - Xena 소프트웨어를 사용하여 비개방형 파일 포맷을 보존 수명을 더 높일 수 있는 개방형 파일 포맷으로 변환하는 것 - 개방형 파일 포맷에 기반한 보존 파일 포맷은 파일을 읽을 때 못 읽을 수도 있는 확률을 줄임 - Xena에서는 Plain text 카테고리에 스타일시트, 데이터베이스, SQL 등을 포함하고 있으며, 파일 보존시 개방형 파일 포맷으로 Unicode 또는 ASCⅡ 코드로의 변환을 추천 • 무결성, 진본성 보장방안 - DPR Processing의 검역-보존-저장단계마다 Checksum checking을 수행해 무결성 · 진본성 보장 - Checksum checking : Manifest가 생성 시점으로부터 모든 디지털 기록은 체크섬을 할당받음 - Checksum checking은 DPR 프로세싱의 전 단계에서 수행됨

스위스	· 스위스 국가기록원(SFA)에서는 Digital Archiving Policy를 통해 데이터세트를 포함한 전자기록물에 대한 지침을 제시 · 데이터세트에 대한 정의를 따로 하고 있지 않으나, 데이터베이스에 저장되는 행정, 과학, 경제 데이터들에 대한 아카이빙을 하고 있음 - 실데이터(Primary data)와 그에 대한 설명데이터(descriptive metadata)를 디지털 아카이빙 범위로 지정 · 아카이빙을 위한 요구사항 : 원본성, 진본성, 무결성, 가용성 · 아카이빙 프로세스 - SFA에서 정의하는 '아카이빙' 과 '아카이빙 프로세스'의 의미는 기록관리를 위한 모든 프로세스를 일컬으며 다음의 5단계로 구분함 - ① 기록물 생산기관에서 필요로 하는 사전 작업을 위한 지원, ② 평가 ③ 습득, ④ 기록물 보호, ⑤ 배포 및 접근 · 아카이빙 전략 - Migration principle을 기본 전제로 하기 때문에 기록물에 대해 필요할 때마다 알맞은 포맷으로 마이그레이션을 수행함으로서 사용성 유지 - 특정 시스템 애플리케이션에 독립적인 아카이빙 전략 - 이관대상 전자기록물은 콘텐츠와 콘텐츠 간 관계정보를 SFA저장소로 전송 · 보존포맷 - 보존에 적합하도록 잘 정의되고, 표준화된 파일 포맷을 보존포맷으로 지정 - 이관 받을 시 지정된 보존포맷으로 제출되었는지 검사 - 기존 보존포맷이 구형화 될 경우 새로운 보존포맷으로 대체할 수 있으며, 기존에 저장되있던 구형 포맷에 대해서는 SFA에서 새로운 보존포맷으로 변환
스웨덴	· ESSArch : OAIS 기반의 디지털 정보 장기 보존 오픈소스 솔루션 · ET(EssArch Tool)와 EPP(ESSArch Preservation Platform)으로 구성 - ET는 로깅 능력이 있는 SIP Package tool로서 준비, 생성, 배달, 수령 단계를 통해 디지털 정보 보존 - EPP는 보존 플랫폼으로 Control area라는 통제된 환경에서 IP들을 관리하고 보관저장소에 저장 · METS, PREMIS 등의 메타데이터를 사용하여 내용물을 설명하고 보존 - METS: XML 문서 포맷으로 모든 IP에서 기술되며, METS를 이용해 SIP패키지나 내용물 설명 - PREMIS: METS와 함께 사용되어 데이터파일을 위한 이벤트나 기술적인 메타데이터를 기술하고, 내용물을 보존 · 보존포맷은 호주 NAA의 Xena 디지털 보존 소프트웨어 사용 · 무결성, 진본성 보장 방안 - 'Diff Check'를 함으로써 EPP내의 로그 정보 확인 - Control Area에서 AIC(Archival Information collection)에 의해 기술된 원본 IP와 비교하여 확인 - 추가, 삭제 또는 업데이트된 파일이 발견될 시 EPP GUI에 나타나며, Work area로 Check-out 되어 재프로세싱

뉴질랜드	· 공공기록 관리 정책(Public Records Act 2005, 이하 PRA)에 의해 정부데이터에 대한 기록관리 · PRA에 의해서 데이터세트도 공공기록물로써 기록관리 · 뉴질랜드 국가기록원에서 데이터세트를 따로 이관 받지 않고, 개별 기관에 아카이빙 지침만 제시하고 개별 아카이빙 · 데이터세트 기록관리 프로세스 - 기록관리를 위한 책임 제시 - 적절한 메타데이터 생성 - 기록관리 대상 데이터세트 복사본 생성 - 데이터 마이그레이션 계획 수립 - 데이터 전송 계획 수립 - 이관 데이터에 대한 보존, 접근 계획 수립 - 보안 관리 계획 수립 - 기록관리 관련 법안 제고 - 기록관리를 위한 요건 확인 · 메타데이터 - 뉴질랜드 국가기록원에서는 전자기록물 기록관리를 위한 메타데이터 표준을 제공 - ISO 23081-1과 AS/NZS ISO 23081-2를 기반으로 작성 - 크게 2가지 메타데이터로 구분됨 : Point of Capture Metadata, Recordkeeping Process Metadata · 무결성 위협 요소 및 보장 방안 - 기록물의 무결성을 위협하는 요소 : 기록물에 대한 검증되지 않은 접근과 사용, 기록물 저장소의 상태, 기록물 관리에 대한 부적합한 행위, 이관 환경에서의 크고 작은 위험 요소, 저장매체의 성능 저하, 기술적 노후화 - 무결성 보장 방안 : 기록물의 수정·손상·삭제로부터 보호, 기록물의 변조를 방지, 기록물의 가용성 보장

(출처 : 국가기록원, 2015 재편집)

<표 9> 국외 행정정보데이터세트 관리 및 보존 전략 특징

1.3.2 국내연구동향

가. 장기보존 정책 및 전략 개요(국가기록원 , 2017)

○ 국가기록원의 국가기록관리혁신추진단은 전자기록관리 체계 재설계를 위해 장기보존 정책, 전자기록 유형별 관리체계 및 포맷정책 재설계 등을 혁신과제로 추진

○ 그 중 전자기록 장기보존 정책의 경우, 전자기록 장기보존에 대한 기본 정책 없이 유형별 및 정보기술별 절차만이 존재. 이에 전자기록의 장기보존 목표, 전략 등 기본 정책을 수립하고자 함

- 세부과제 7-1 : '장기보존을 위한 기본 정책 수립'
- 세부과제 7-2 : '전자기록 유형별 관리 및 보존 설계 및 제도화'

○ 현재 혁신과제 이행과정 중 중간보고 내용(국가기록원, 2018)

- 전자기록 장기보존 정책 수립
  - · 전자기록 장기보존 기본정책 수립
  - · 전자기록 장기보존 상세 이행계획 수립
  - · 전자기록 장기보존 정책 및 이행 점검 체계 마련
  - · 전자기록 장기보존 정책과 이행 지원을 위한 연구기능 강화
- 전자기록 유형별 관리체계 및 포맷정책 재설계
  - · 전자기록물 유형별 관리체계 및 포맷정책의 기본방안 수립
  - · 전자기록물 유형별 관리체계 재설계
  - · 전자기록물 유형별 통합 수집·관리를 위한 시스템 고도화
  - · 전자기록 유형별 보존포맷(안)이 검토 및 향후 검토 필요사항으로 등장
  - · 파일형(NEO 1.0, 2.0, VEO 1.0, 2.0, XENA), 폴더형(VEO 3.0, Bagit, DSpace)

<table>
<tr><th colspan="3">구분
유형</th><th>문서보존포맷 1)
Object(File format)</th><th>장기보존포맷 2)
Packaging(encapsulation)</th></tr>
<tr><td colspan="3">표준전자문서</td><td>· PDF/A, DOCX, ODT</td><td rowspan="7">• 파일형, 폴더형 취사선택 가능
</td></tr>
<tr><td colspan="3">데이터세트</td><td>· SIARD, CSV, JSON, XML, XLSX, ODS</td></tr>
<tr><td rowspan="4">기타</td><td rowspan="3">시청각</td><td>이미지</td><td>· JPEG(.jpg, .jpeg), JPEG2000(.jp2), TIF, TIFF</td></tr>
<tr><td>음성</td><td>· FLAC, MP3, MP4, WAV</td></tr>
<tr><td>영상</td><td>· FFV1, MP$</td></tr>
<tr><td colspan="2">웹기록</td><td>· KWARC</td></tr>
<tr><td colspan="2">SNS,이메일 등</td><td colspan="2">· '18년 대통령기록관 R&D 결과 활용
(API 방식, SNS플랫폼 자체 아카이빙, 제3자 아카이빙 서비스 중 검토·선택)</td></tr>
</table>

(출처 : 국가기록원, 2018)

<표 10> 전자기록 유형별 보존포맷(안)

○ 장기보존 전략의 경우, 표준전자문서의 장기보존포맷 변환 외에는 부재한 것으로 분석됨

1) 문서 생산 당시의 애플리케이션이 없어도 재현을 가능하도록 하는 포맷
2) 장기간 진본성 및 무결성 보장을 위하여 원천기록과 함께 문서보존포맷, 메타데이터, 전자서명 등을 하나로 묶어 구성한 포맷

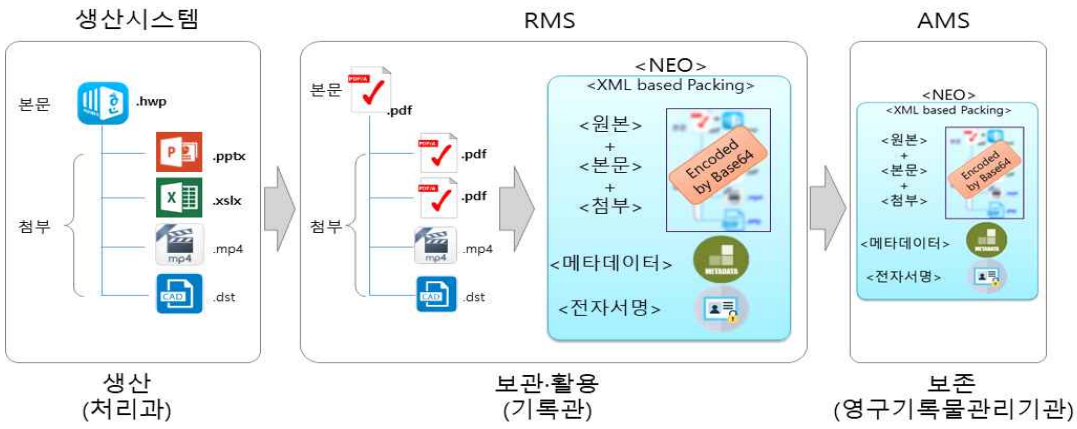
나. 장기보존기술 개요(국가기록원, 2019)

○ 법적근거

- 「공공기록물 관리에 관한 법률」 제20조(전자기록물의 관리) 제1항 제1호
- 동법 시행령 제36조(기록관 및 특수기록관의 전자기록물 보존)
- 동법 시행령 제40조(기록관 및 특수기록관의 소관 기록물 이관)
- NAK 30:2008(v1.0)「전자기록물 문서보존포맷 기술규격」
- NAK 31:2017(v2.1)「전자기록물 장기보존포맷 기술규격」

○ 장기보존포맷변환 프로세스

- 생산시스템에서 RMS에 등록할 때 오피스 계열 파일들은 PDF/A로 변환하고 NEO 패키징을 수행



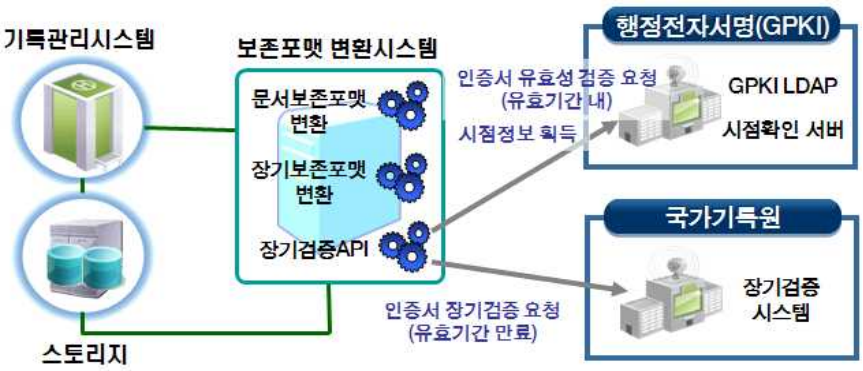
<그림 8> 장기보존포맷 변환 프로세스

○ 장기보존포맷 변환 검증

- 장기보존포맷 기록물의 경우 변환에 사용된 전자서명의 유효기간이 만료되면 국가기록원 장기검증시스템을 통해 기록물의 유효성 검증 가능
- 변환 전 점검사항
  - · 시스템 운영환경 : 최소 사양 이상의 시스템 준비
  - · 네트워크 환경 : 장기보존포맷 변환을 위한 인증서 검증 등을 위해 각 기관과 국가기록원간 네트워크 구간의 방화벽 포트 허용 필요

구분	사양
서버	· (최소) NT장비, 2CPU 이상(Xeon 3.0GHz), 메모리 4GB ※ 변환대상 전자기록물 규모에 따라, 성능 추가 및 여러 대의 병렬설치 가능
운영체제	· Window 2003 이상(32bit 이상)
스토리지	· 문서보존포맷 및 장기보존포맷 최초 변환 시 대상 파일용량의 약 3.5배 이상 필요

<표 11> 보존포맷변환을 위한 시스템 운영환경 조건



<그림 9> 보존포맷변환시스템 업무 구성

다. 문서보존포맷

○ PDF/A

- 국가기록원에서 정한 보존기간 10년 이상인 전자기록물에 대한 문서보존포맷
  · 편재성, 안정성, 메타데이터지원, 상호운영성, 진본성, 표현력, 검색 기능에서 우수
  · 국가기록원은 현재 PDF/A 기반 표준전자문서 시스템으로 최적화 되어 있음

	XML	Text	이미지 (TIFF/BITMAP)	PDF	CSD	PDF/A
공개용 표준	상	상	상	하	하	상
편재	상	상	상	중	중	상
안정성	상	하	상	상	상	상
메타데이터 지원	상	하	하	상	상	상
상호운영성	상	상	상	상	상	상
진본성	상	상	상	상	상	상
처리능력	상	하	중	상	상	상
표현력	중	하	상	상	상	상
검색 기능	상	상	하	상	상	상

<표 12> 다양한 문서 포맷 비교표 - PDF/A

○ 문서보존포맷 변환 소요시간

- 변환 불가 파일 유형은 변환 제외
- 변환대상 파일 크기, 문서 페이지 수에 따라 변환소요시간 비례
- 통합형/공동형 표준기록관리시스템의 경우 서버를 공유하므로 변환시간이 장시간 소요될 가능성이 높음
- 시스템 환경에 따른 적절한 프로세스 수 설정

변환 시스템 테스트 환경			파일 건당 평균변환시간 * 전체 파일수 : 1,382개 (기록물건수 : 870건)
시스템사양	대수	서버당 변환 프로세스 수	
저사양 CPU(Intel Xeon 4Core 2Ghz) MEM(2G) OS(win2003 enterprise)	1대	1	9.06초
		4	2.10초
고사양 CPU(Intel Xeon 32Core 2.4Ghz) MEM(32G) OS(win2012 enterprise)	1대	1	4.37초
		8	0.92초

<표 13> 시스템 환경별 문서보존포맷 변환소요시간

라. 장기보존포맷 기술 및 표준 현황

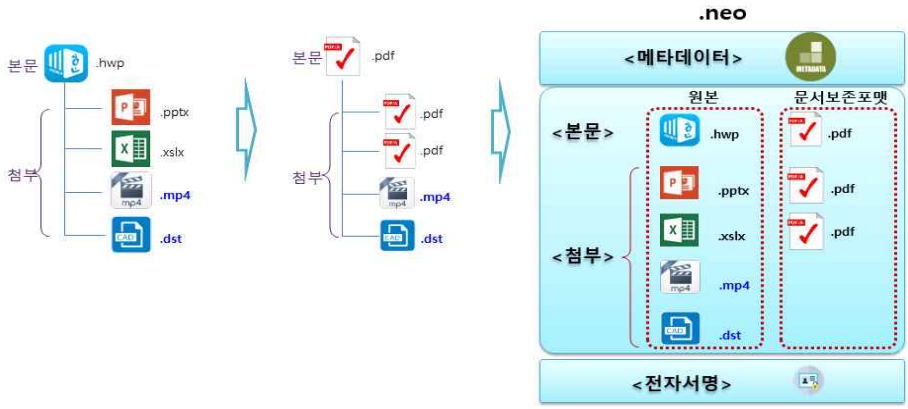
○ NEO(NAK Encapsulated Object)

- 국가기록원에서 정의한 장기보존포맷 스키마에 따라 XML 형태의 텍스트 파일(.neo) 안에 패키징하는 전자기록물 장기보존포맷 기술규격

· 하나의 XML 파일 안에 포함시키는 패키지 방식

· 메타데이터 + 파일인코딩(Base64) + 전자서명(GPKI)

· 비대칭 전자서명(공개키/개인키, 공인인증기관)



<그림 10> 장기보존포맷 NEO 구조

○ 전자기록물 문서보존포맷 기술규격 (NAK 30 : 2008(v1.0))

- PDF/A-1

· 문서보존포맷 규격("A" 는 Archive, "1" 는 인쇄가 가능한 모든 매체를 대상으로 하고 있음을 의미함)

· PDF/A-1a : 적합성 A 수준(ISO 19005-1의 모든 기술규격을 수용)

· PDF/A-1b : 적합성 B 수준(ISO 19005-1에서 유니코드 문자맵과 논리적 구조를 제외한 모든 조건을 만족)

- PDF/A-2

· 동영상, 오디오, 3D 그래픽, JPEG-2000압축 등에 대한 보존포맷

· 문서의 장기보존에 위배될 가능성이 있는 일부 요소(암호화, 내장파일, LZW압축, 투명성, 멀티미디어, 자바스크립트)를 금지하여 사용

고려요소	내용
공개용표준	· 특정업체가 독점적으로 소유권을 가지고 있지 않아야 함 · 누구나 참조하고 이용할 수 있게 공개되어야 함 · 사용자에게 경제적으로 영향을 주지 않아야 함
편재성	· 오랜 기간 동안 사용될 가능성이 있어야 함 · 많은 곳에서 사용되는 포맷이어야 함 · 포맷의 이용에 대해 지역이나 기간의 제한을 받지 않아야 함
안정성	· 원래 생산된 문서를 문서보존포맷으로 변환할 때 원래 문서의 내용, 구조, 맥락정보 등을 훼손시키지 않고 보존해야 함 · 문서의 내용이 시간이 지남에 상관없이 그대로 유지될 수 있어야 함 · 버전의 지속적 변화에 상관없이 호환성이 유지되어, 구 버전의 문서 또한 그대로 볼 수 있어야 함

메타데이터 지원	· 문서를 장기보존 할 때 필요한 메타데이터를 지원해야 함
상호운용성	· 운영체제나 플랫폼에 독립적이어서 서로 다른 시스템 간에 문서의 마이그레이션을 쉽게 하여 한 기관에서 생성된 문서보존포맷은 다른 기관 혹은 외부 이용자도 사용할 수 있어야 함
진본성	· 문서의 내용, 외형 등이 시간의 경과에 상관없이 원래의 모습과 일치하며, 생산자가 생산하려고 했던 원래의 취지에 맞는 그 전자문서라는 증명이 가능하도록 문서가 훼손, 위조, 변조가 되지 않도록 하는 포맷이어야 함
표현력	· 원래 생산된 전자문서의 내용뿐만 아니라 문서의 외형과 구성 그 자체가 그대로 표현되어야 하며 복원될 수 있어야 함 · 문서의 진본성과도 연결되는 조건으로서 원문적 특성이 원래의 문서 그대로 보존될 수 있어야 함
검색기능	· 문서 내부에서 이용자가 원하는 문서내용에 대한 검색 기능을 제공해야 함

<표 14> 문서보존포맷 고려요소

○ 전자기록물 장기보존포맷 기술규격 (NAK 31 : 2017(v2.1))

- 장기보존포맷 고려사항

· 자체충족성 : 기록물이 생산된 당시의 내용을 그대로 재현하여 읽고 이해할 수 있도록 하기 위하여 전자기록물은 시스템, 외부데이터 등에 독립적이어야 함

· 자체문서화 : 전자기록물은 자체적으로 기록물과 관련된 기술(記述)과 맥락에 대한 이해를 줄 수 있는 정보를 포함하여, 미래의 이용자들이 장기 보존된 기록물의 내용을 이해할 수 있도록 해야함

· 구조화된 텍스트 인코딩 : 캡슐화하는 기록물은 바이너리 데이터보다는 구조화된 텍스트 형식으로 인코딩되어야 함

· 무결성 : 전자기록물이 위조 또는 변조 되지 않았음을 보장하여 기록물의 법적증거를 확보하여야 하며, 무결성을 보장하기 위해 인증정보를 적용해야 함

- 장기보존포맷 구성요소

· 원문 : 생산자가 생산 또는 접수한 전자기록물 원본(진본)으로 진본성을 보장하기 위해 포함

· 문서보존포맷 : 문서 생산 또는 접수 당시의 애플리케이션이 없이도 해당 문서의 내용과 외형을 그대로 재현한 포맷으로, 시간과 기술변화에 상관없이 이용자가 문서 내용에 접근할 수 있게 함

· 장기보존포맷 메타데이터 : 기록물의 생산 접수부터 관리·보존에 이르는 전 과정을 기술(記述)한 정보로, 기록물 생애주기 전 기간에 걸쳐 진본성, 신뢰성, 무결성, 이용가능성을 보장하며, 기록물을 관리하고 보존·이해할 수 있도록 지원함

· 전자서명 : 전자기록물의 진본성 및 무결성 보장을 위해 장기보존포맷에 포함되는 정보로, 전자서명은 인증정보와 잠김인증정보로 구성되어 있음

마. 행정정보데이터세트 관리 및 보존 전략 현황

○ 전자기록관리 관련 정보시스템 현황(국가기록원, 2018)

- 행정정보시스템은 중앙부처, 지자체, 공공기관 등 각 기관이 고유·공통 업무 수행을 위해 총 17,973개('16년 12월 기준) 구축·운용되고 있음

○ 행정정보데이터세트에 대한 장기보존

- 행정정보데이터세트에 대한 장기보존은 실제로 시행되고 있지는 않지만 장기보존 정책·전략·기술에 대해 지속적인 연구 및 방안이 추진되고 있음

	중앙부처	지자체	공공기관
기관 고유 업무	1,408	5,617	4,315
기관 공통 업무	280	1,159	890
시스템관리, 업무지원	519	2,140	1,645
합계	2,207	8,916	6,850

<표 15> 행정정보시스템 운영현황

구 분	산림자원 통합관리시스템	국민신문고 시스템	전자연구노트 시스템	특허넷	국토정보 시스템	화학물질종합 정보시스템
운영기관	산림청	국민권익 위원회	한국과학 기술원	특허청	국토교통부	화학물질 안전원
DB 크기	600M	1.2T	2T	15T	3T(원천데이터) 400G(DW/DM)	329G

<표 16> '17.7월 행정정보시스템 내 데이터세트 현황조사 결과

유형구분	행정정보시스템
데이터베이스 가치에 따른 정보시스템 구분	· 인덱스/검색도구 · RMS(Record Management System) · 통계데이터베이스(Statistical Databases) · 지원시스템(Support System)
조직의 기능에 따른 구분	· 조직레벨 시스템, 지식레벨 시스템, 관리레벨 시스템, 전략레벨 시스템
데이터의 처리에 따른 구분	· EDMS(Electronic Documents Management System) · OLTP(On-Line Transaction Processing)
데이터의 범위에 따른 구분	· 공동자원 유형, 내부자원 유형
데이터의 성격에 따른 구분	· 데이터 처리 시스템, 데이터 집계 시스템, 데이터 관리 시스템, 데이터 검색/서비스 시스템

(출처 : 국가기록원, 2015)

<표 17> 행정정보시스템 유형

제2장 총괄연구개발과제의 최종 연구개발 내용 및 방법

1. 연구개발 내용

○ 데이터세트 유형 전자기록 현황 및 장기보존기술 조사 및 분석

- 데이터세트 유형 전자기록 관련 마이그레이션 기반 장기보존기술 조사 및 분석

(※ SIARD(Software Independent Archiving of Relational Databases) 2.1 등)

- 데이터세트 유형 전자기록 관련 에뮬레이션 기반 장기보존기술 조사 및 분석

(※ 아카이브 기관 및 다른 분야 현황(예, 디지털포렌식, 게임, 시스템 마이그레이션 등))

- 공공기관의 행정정보시스템에서 생산・관리되는 데이터세트 형태・운영・관리 현황

○ 데이터세트 유형 전자기록 현황 및 장기보존기술 조사 및 분석

- 데이터세트 유형 전자기록의 보존포맷 선정기준 수립 및 선정기준에 따른 포맷 제시

(※공통기준 중 반드시 준수해야 하는 필수사항과 선택사항을 구분하여 제시)

- 고유기준의 경우, 진본성 보장을 위해 반드시 유지되어야 하는 디지털 자원의 주요 특성(내용, 맥락, 외관, 구조, 기능, 기술/재현속성 등의 측면)을 포괄하는 기준 제시

○ 다양한 DBMS 대상으로 데이터세트 유형 전자기록 보존포맷 변환 검증 시험 및 국산 DBMS 대상 변환 기능 개발

- 공공기관에서 주로 사용하고 있는 DBMS들을 대상으로 문서보존포맷 변환 검증을 위한 테스트베드 구축 (큐브리드, Oracle, MySQL, SQL Server 등)

- 테스트베드 기반 보존포맷 변환 검증 시험

- DBMS 외부 파일시스템에 저장되는 파일 처리 과정 분석 및 방안 제시

- 국산 DBMS 큐브리드를 대상으로 보존포맷 변환 기능 개발

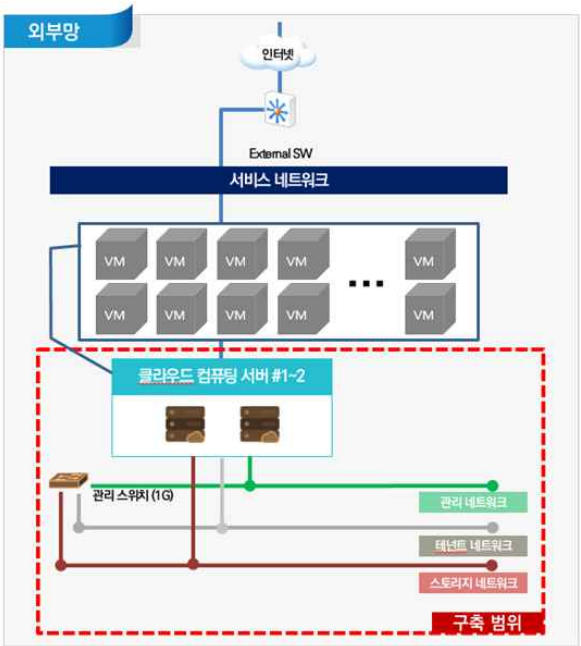
- 데이터세트의 규모, 환경 등에 대한 확인 후 가능성 제시

- 검증과정 및 국산 DBMS 대상으로 보존포맷 변환 SW 개발시 발생 문제점 장·단점 등을 조사·분석하여 이를 반영한 보존포맷 제안

<그림 11> 마이그레이션 연구 내용 및 방법 개요

○ 에뮬레이션 방식에 따른 기술적합도 검증을 위한 테스트베드 구축

- 에뮬레이션 시험 검증을 위한 하드웨어 시스템 구축



예정 장비 스펙

구분	사양	수량	비고
클라우드 컴퓨팅 서버	• CPU: 24C/48T • RAM: 256GB • DISK1: SSD 480GB * 2EA • GlusterFS DISK: 1TB * 6EA	2대	GlusterFS
관리 스위치	• 1G 8Port Ethernet	1대	

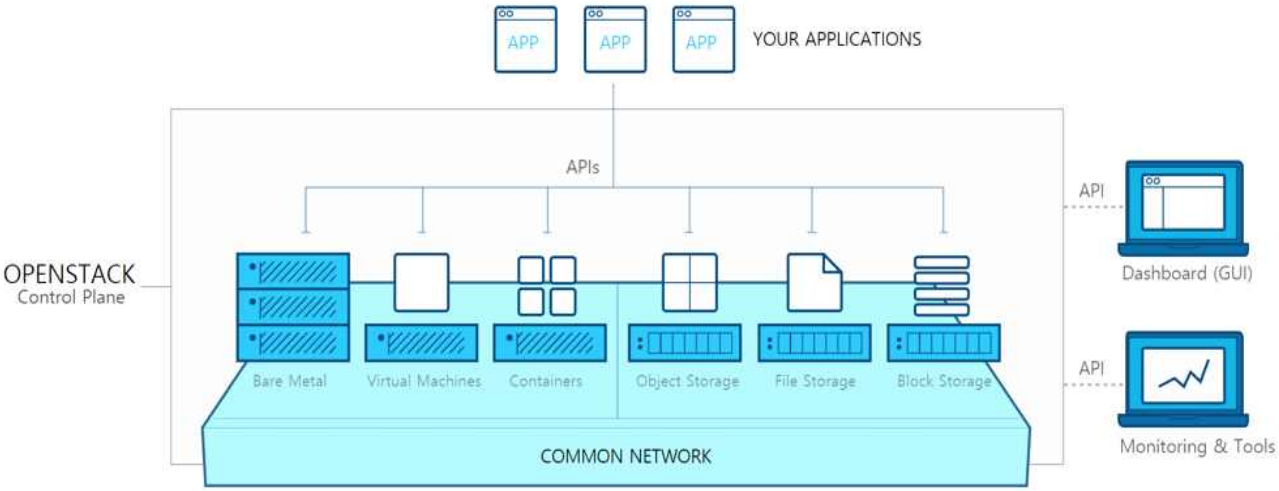
범례

External SW · 인터넷 · 클라우드 컴퓨팅 서버 · L2 SW

서비스 네트워크 (1G) · 스토리지 네트워크 (1G) · 관리 네트워크 (1G) · 테넌트 네트워크 (10G)

<그림 12> 테스트베드 구성 개요 및 장비 스펙(안)

◦ 오픈스택(OpenStack) 기반 클라우드 인프라 환경 구축



<그림 13> 클라우드 인프라 환경 기반 에뮬레이션 시스템 개요

○ 데이터세트 유형 전자기록의 에뮬레이션 시험

- 선정된 데이터세트의 에뮬레이션 보존 방식 테스트
- 선정된 데이터세트의 에뮬레이션 후 원천 데이터세트와의 정합성 검증항목 마련 및 점검

○ 테스트베드 구축·시험 결과에 따른 보존적합방식 제안

- RDB와 Non-RDB에 따른 보존포맷 제안
- RDB의 경우, 시스템・SW 환경, 시스템・SW 연계상황 등에 따라 보존방안 제안
- 공공기관의 데이터세트 유형 전자기록 보존 및 활용을 위한 지침 작성 및 제출

2. 연구개발 방법

○ 데이터세트 유형 전자기록 현황 및 장기보존기술 조사

- 데이터세트 유형 보존포맷 조사 (SIARD 2.1 표준, SIARD Suite, SIARD 해외사례, SIARD 라이선스 등)
- 공공기관 행정정보시스템의 생산・관리되는 데이터세트 형태・운영・관리 현황

○ 데이터세트 보존포맷 선정체계 수립 및 보존포맷 선정

- 데이터세트 유형 전자기록의 보존포맷 선정기준 수립
- 데이터세트 유형 전자기록의 보존포맷 선정기준에 따른 보존포맷 제시

○ 국산 DBMS 큐브리드 대상 보존포맷 변환 · 복원 기능 개발

- '큐브리드 JDBC Driver' 정합 및 '큐브리드 JDBC Wrapper' 작성
- Routine Type 변환 · 복원 기능 보완
- 큐브리드 고유의 기능 및 요소 추가 확장

○ 데이터세트 보존포맷 변환 · 복원 검증

- Oracle, MySQL, SQL Server, 큐브리드 4종의 DBMS 대상으로 보존포맷 변환 · 복원 검증 시험
- DBMS의 기능, 데이터, 크기 등에 대해 변환 · 복원 검증 시험
- Oracle, 큐브리드 2종 DBMS의 실제 데이터를 대상으로 변환 · 복원 검증 시험

○ 보존방식에 따른 기술적합도 검증을 위한 테스트베드 구축

- 에뮬레이션 시험 검증을 위한 하드웨어 시스템 구축
- 오픈스택 기반 클라우드 인프라 환경 구축

○ 데이터세트 유형 전자기록의 에뮬레이션 시험

- 선정된 데이터세트의 에뮬레이션 보존 방식 테스트
- 선정된 데이터세트의 에뮬레이션 후 원천 데이터세트와의 정합성 검증항목 마련 및 점검

○ 테스트베드 구축·시험 결과에 따른 보존적합방식 제안

- 데이터세트 유형, 시스템 구성 운영 형태 등에 따라 마이그레이션 또는 에뮬레이션 제시 절차 및 방안 제안
- 마이그레이션 방안 적용 시, 선정된 데이터세트 보존포맷으로 변환 · 보존 · 검증 · 활용을 위한 지침 작성
- 에뮬레이션 방안 적용 시, 실험된 클라우드 기반 에뮬레이션 방식으로 변환 · 보존 · 검증 · 활용을 위한 지침 작성

제3장 총괄연구개발과제의 최종 연구개발 결과

1. 데이터세트 유형 전자기록 보존포맷 선정 및 테스트베드 구축 · 시험 · 검증

○ 데이터세트 유형 전자기록 현황과 장기보존기술 조사(마이그레이션 중심으로)

- 데이터세트 유형의 장기보존기술 중 현재 가장 많이 논의되고 있는 SIARD 2.1 표준에 대해 조사함
- SIARD 2.1 표준은 메타데이터와 테이블 데이터가 결합된 구조로 원본 DBMS를 사용할 수 없게 되더라도 데이터세트의 내용을 확인할 수 있는, 독립적인 단일 파일 생성이 가능한 표준임
- SIARD 2.1 표준을 따르는 SIARD Suite은 오픈소스 프로젝트이기 때문에 오픈소스 프로젝트 라이선스 정책에 대한 조사도 병행함
- 각 라이선스 정책 별로 공개 범위가 상이하나 공익적 차원에서 수정하여 재배포하는 경우에도 이를 공개하는 것이 바람직하다고 판단 됨
- SIARD는 스위스, 덴마크, 포르투갈을 중심으로 개발, 활용되고 있음

○ 데이터세트 보존포맷 선정체계 수립 및 보존포맷 선정

- 기록물 생산 환경 및 정보기술의 변화로 다양한 유형의 전자기록이 지속적으로 증가하고 있는 상황에서 일괄적으로 적용하는 현행 단일 문서보존포맷과 장기보존포맷 전략으로는 대응하는 데 한계가 있음
- 다양한 기록유형 및 기술변화 등을 고려하여 전자기록 장기보존의 지속가능성, 유연성, 확장성, 안전성 등을 확보할 수 있는 전자기록 장기보존전략을 개발할 필요가 있음
- 특히, 보존포맷과 관련하여 모든 기록유형에 적용 가능한 보존포맷 선정 기준(공통기준)과 기록유형별 특성을 고려한 보존포맷 선정 기준(고유기준)을 마련함으로써 보존포맷 다양화 전략이 필요
- 이를 위해, 모든 유형에 적용 가능한 보존포맷 선정 기준(공통기준)과 데이터세트 보존포맷 선정 기준(고유기준)을 제안하였으며, 이를 기준으로 보존포맷 적합성 평가체계 개발
  · 공통기준 : 상위기준 총 5개, 하위기준 총 10개 / 데이터세트 고유기준 : 3개
- 본 연구에서 개발한 보존포맷 적합성 평가체계를 적용하여 데이터세트 보존포맷인 'SIARD' 평가검증
  · 데이터세트 보존포맷 적합성 평가 결과 : 'SIARD' - C등급(양호)

○ 국산 DBMS 큐브리드 대상 보존포맷 변환기능 개발

- 주관기관과 협의한 결과, 현재 G클라우드 표준으로 등록되어 있는 국내 DBMS인 큐브리드를 대상으로 SIARD Suite에 확장한 보존포맷 변환기능을 진행하여 개발을 완료함
- 큐브리드 DBMS를 SIARD Suite에 확장 도입하기 위한 구현 방향을 소개함
- SIARD Suite을 구성하는 오픈소스코드 중 어떤 부분을 수정해야 하는지를 소개함
- 실제 구현 과정과 구현한 소스코드에 대해 설명함

○ 데이터세트 유형 전자기록 보존포맷 변환 · 복원 검증

- 데이터세트 유형 전자기록 보존포맷(SIARD)의 적합성 여부 판단을 위해 검증 시험 진행
- SIARD 변환 대상 DBMS는 기관 협의를 통해 4종(MySQL, SQL Server, Oracle, 큐브리드)의 DBMS를 선정함
- 기존의 SIARD는 국산 DBMS인 큐브리드를 지원하지 않기 때문에 SW 개발을 통해 큐브리드를 지원하는 버전의 SIARD를 사용하여 검증 시험을 진행함
- 4종의 DBMS ↔ SIARD 시험을 “사전 시험”, “본 시험”, “DB Size 시험”으로 진행함
- “사전 시험”은 SIARD에서 제공하는 기본적인 변환 기능에 대한 검증을 목적으로 진행하였으며, 사전 시험을 통해 4종 DBMS의 Data Type, Key Type, Routine Type 중 SIARD 포맷으로 변환이 가능한 부분과 불가능한 부분을 도출함
- “본 시험”은 DBMS의 Data와 SIARD 포맷으로 변환된 Data들에 대해 원본DB와 업로드DB의 동일 여부 검증하였으며, 변환 검증 시험의 Data 동일 여부에 관한 검증은 DBMS 종류에 따라 TOAD Data Point, 큐브리드 MANAGER에서 제공하는 비교 마법사 TOOL을 사용하여 검증을 진행함
- “DB Size 시험”은 DB SIZE 별도 Download 및 Upload의 시간 및 변환 여부 검증함
- 2개 종류의 실데이터(정부청사관리본부 큐브리드 DB와 국가기록원에서 제공한 오라클 DB)를 대상으로 보존포맷 변환·복원 검증 수행
- 약 1GB의 큐브리드 DB는 모든 데이터가 잘 변환되었으며, CUBRID Manager에서 제공하는 데이터 비교 도구를 활용하여 진행함
- 약 1.5GB의 오라클 DB는 TIMESTAMP 타입(시분초 데이터는 업로드DB에 포함안됨)을 제외한 모든 데이터가 잘 변환·복원되었으며, TOAD Data Point 도구를 활용하여 진행함
- TIMESTAMP는 추가적인 SW개발을 통해 모든 데이터가 잘 변환·복원되었음을 확인함

○ 테스트베드 구축 · 시험 결과에 따른 장기보존방안

- 큐브리드 지원하기 위해 SIARD Suite를 확장 및 개발하고, 4개 DBMS에 대한 검증 수행

한 결과를 분석하여 장기보존방안을 제시함

- 먼저 SIARD Suite이 가지고 있는 장단점을 분석하고, 이를 기반으로 향후 SIARD 표준과 SIARD Suite을 활용하여 데이터세트 보존에 어떻게 활용할지를 제시함

- 또한, 향후 SIARD Suite확장하여 다른 DBMS를 지원하고자 할 때, 어떤 단계로 진행해야 하며 어떤 부분을 수정 및 보완해야 하는지 제시함

2. 클라우드 기반 전자기록의 장기보존기술개발 테스트베드 구축 및 에뮬레이션 시험 · 검증

○ 보존방식에 따른 기술적합도 검증을 위한 테스트베드 구축

- 에뮬레이션 시험 검증을 위한 하드웨어 시스템 구축

· 에뮬레이션 시험 검증 요구사항에 따라 클라우드 인프라 규모 사이징, 하드웨어 아키텍처 및 네트워크 구성(안) 수립

· 기존 클라우드 인프라 구축 경험을 통해 안정성이 검증된 하드웨어 장비 확보

· 원활한 연구개발 추진을 위해 하드웨어 시스템을 민간 데이터센터에 구축

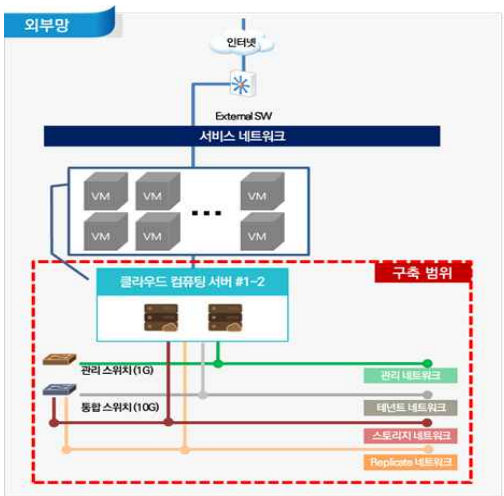
- 오픈스택 기반 클라우드 인프라 환경 구축

· 안정성이 검증된 오픈스택 퀸즈 버전으로 클라우드 인프라 환경 구축

· 에뮬레이션 방식의 전자기록물 보존 시험·검증을 위해 가상머신, 도커/컨테이너 환경 제공 수준의 클라우드 인프라 환경 구축

· 다양한 오픈스택 프로젝트 중 불필요한 프로젝트를 제외한 최적의 프로젝트들만 설치

- 에뮬레이션 시험 검증을 위한 하드웨어 시스템 구축

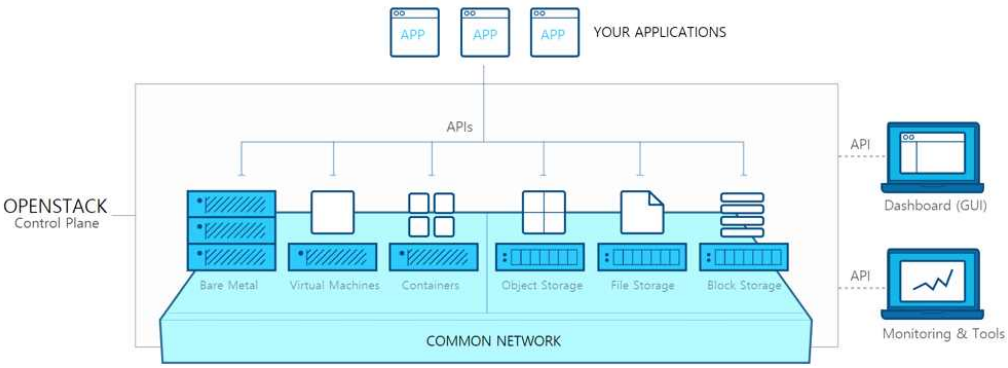


예정 장비 스펙

구분	사양	수량	비고
클라우드 컴퓨팅 서버	• CPU: 24C/48T • RAM: 256GB • DISK1: SSD 480GB * 2EA • GlusterFS DISK: 1TB * 6EA	2대	GlusterFS
통합 스위치	• 10G 12Port Ethernet	1대	
관리 스위치	• 1G 8Port Ethernet	1대	

<그림 14> 에뮬레이션 시험 검증을 위한 하드웨어 시스템

- 오픈스택(OpenStack)기반 클라우드 인프라 환경 구축



<그림 15> 오픈스택(OpenStack)기반 클라우드 인프라 환경 구축

프로젝트명	서비스 내용
glance	· 이미지 서비스
horizon	· 포탈 서비스
keystone	· 인증서비스
neutron	· 네트워크 서비스
nova	· 컴퓨트 서비스
heat	· 오케스트레이션 서비스
magnum	· docker 서비스

<표 18> 오픈스택 프로젝트 구성

오픈스택잇 UX 특징

- 클라우드잇 디자인 노하우에 기반한 UX(사용자 경험) 극대화
- 위자드를 통한 자원 생성 화면 제공
- 목록과 상세내용 동시 표현
- 화면 이동을 최소화한 구성
- 관리자/사용자 권한에 따른 서로 다른 기능 및 화면 구성 제공
- 사용자를 위한 기능을 별도로 제공하는 셀프 서비스 포털 제공

<그림 16> 오픈스택잇(OpenstackIt) 특징

제4장 총괄연구개발과제의 연구결과 고찰 및 결론

4.1 데이터세트 유형 전자기록 현황과 장기보존기술 조사

○ SIARD 2.1 표준은 메타데이터와 테이블데이터가 결합된 장기보존기술

- 원본의 데이터베이스 소프트웨어를 사용할 수 없게 되더라도 XML과 SQL:2008 표준에 기반하여, 데이터베이스 데이터의 접근과 교환을 가능케 할 수 있음

- 국외에서는 SIARD-DK 등 파생형이 활용되고 있으며, 세금계산서 등 실제 행정정보의 장기보존에 적용하는 연구가 진행 중

○ SIARD는 오픈소스 프로젝트이기 때문에 개발 및 배포 시 공개범위에 대한 검토 필요

- 행정정보데이터세트의 장기보존은 공공의 성격을 가지기 때문에, SIARD의 수정된 소스 코드를 공개하는 것이 적합하다고 고려됨

4.2 데이터세트 보존포맷 선정체계 수립 및 보존포맷 선정

○ 파일포맷의 특성과 해당 기록유형의 특성을 근거로 보존포맷 선정기준 개발

- 공통기준은 파일포맷의 특성을 고려하고, 고유기준은 기록유형의 특성을 고려함

○ 보존포맷 선정 기준에 따라 평가 지표 개발

- RDB형 보존포맷: SIARD, Non-DB형 보존포맷 : BIFF8, CSV

○ 다양한 유형의 전자기록에 대한 장기보존의 유연성 확보

- 전자기록 보존포맷 선정을 위한 기준을 마련하여 장기보존의 유연성 확보

○ 다양한 유형의 전자기록에 대한 보존포맷의 확장성 확보

- 전자기록 보존포맷의 평가지표 개발을 통해 보존포맷 다양성 확보

○ 다양한 유형의 전자기록 보존포맷 선정을 위한 고유기준 및 평가지표의 확장 필요

- 다양한 유형의 파일포맷 평가를 통해 평가지표를 지속적으로 보완

- 다양한 유형의 전자기록에 적합한 보존포맷 선정을 위한 고유기준 및 평가지표 보완

4.3 국산 DBMS 대상 보존포맷 변환기능 개발

○ 변환기능 개발에 대한 고찰

- 단순히 JDBC를 확장만으로는 해당 DBMS의 모든 기능과 요소 전체 모두를 Download

및 Upload 할 수 없었음

- 확장하기 위한 DBMS의 기능과 요소들을 Data Type(기본/특수), Key Type, Routine Type으로 구분하여 정리한 다음, SQL:2008 표준, JDBC API의 기능 명세를 함께 분석하면서 가능한 SQL:2008에 포함되도록 JDBC에 수정하거나 기능을 추가해야 함

- 만약 JDBC에서 수용할 수 없는 상황(큐브리드의 경우, Stored Procedure/Function)이라면 별도의 구현 과정이 필요함

4.4 데이터세트 유형 전자기록 보존포맷 변환 검증

○ 2차례에 걸친 보존포맷 변환 검증 시험을 통해 SIARD 포맷 변환, 복구 검증 진행

○ 검증 결과, SIARD는 Routine Type을 제외한 대부분 항목의 변환, 복구가 가능

○ 하지만 Routine Type, MySQL의 “JSON”, Oracle의 “UROWID”와 같이 SIARD 변환이 지원되지 않는 항목과 안정성 확보를 위한 추가적인 개발이 필요

○ 내부 검증 4종의 DBMS(MySQL, SQL Server, Oracle 큐브리드)를 대상으로 진행한 검증 시험 결과, Data Type, Data, PK, FK 등 관계형 데이터세트의 보존에 중요한 정보 및 항목들을 SIARD 포맷으로 안정적인 변환 및 보존이 가능함

※ Oracle의 PK, FK 경우는 SIARD 포맷의 문제가 아닌 Oracle DBMS의 구조적인 문제로 판단

○ 서로 다른 DBMS로 Download 및 Upload를 진행해도 SIARD 파일 내의 Data가 정상적으로 변환, 보존이 되기 때문에 보존포맷으로서 SIARD 포맷의 기능은 출중하다고 판단

○ 하지만 대규모 DB를 SIARD 포맷으로 변환할 경우, 많은 시간이 소요되고 변환 도중 문제가 발생하면 처음부터 변환 작업을 해야한다는 단점이 존재함

○ SIARD 포맷은 구조적으로 안정성을 확보할 수 있는 기능을 추가적으로 개발하는 것이 필요하다고 판단

○ 또한, 검증 시험을 통해 최소 7GB, 180만 개 규모를 가진 DB는 안정적으로 SIARD 포맷으로 변환이 가능하다고 판단함

4.5 실데이터에 대한 보존포맷 변환 검증

○ 큐브리드 DBMS 대상 실데이터 검증 결과 고찰

- 큐브리드에 대한 모든 기능과 요소들이 Download 및 Upload 될 수 있도록 SW를 개발하였기 때문에 DBMS 대상 검증 시험 결과는 원본DB와 복원DB 모두 동일하다고 나왔음

○ Oracle DBMS 대상 실데이터 검증 결과 고찰

- DATE Type의 데이터가 일부 손실되어 추가적인 SW개발이 진행되었음

- Oracle DBMS에서 제공하는 기능 및 요소가 다른 모든 DBMS의 합집합 모두 크기 때문에 이 Oracle DBMS의 모든 기능과 요소를 Download 및 Upload 할 수 있으면 다른 DBMS로 확장할 때 많은 도움이 될 것으로 판단됨

4.6 테스트베드 구축 · 시험 결과에 따른 장기보존방안

○ SIARD를 데이터세트 보존포맷으로 활용하기 위해서는 오픈소스인 SIARD Suite를 수정 보완해야 함

○ 특히, 다양한 DBMS 지원이 필요하며, 새로운 DBMS 지원을 위해 수정해야 할 부분과수정 및 보완하는 과정을 설명함

○ SIARD Suite에서 제공하는 Type과 해당 DBMS의 JDBC에서 제공하는 Type을 (1) 기본 Data Type, (2) 특수 Data Type, (3) Key Type, (4) Routine Type으로 분류

○ 각 분류 별로 지원가능 여부를 조사 및 분석하여 “JDBC 단순 Type 매핑”, “JDBC Type 매핑 및 별도 처리”, “JDBC 새로운 Type 및 처리”인지를 구분하여 구현을 진행함

4.7 클라우드 기반 전자기록의 장기보존기술개발 테스트베드 구축 및 에뮬레이션 시험 • 검증

가. 기대효과

○ (기술적) 클라우드 기반 전자기록 보관 시스템 관련 핵심 기술 확보

- 클라우드 기반의 전자기록 저장 시스템 구성을 통한 안정적인 환경 및 보존 기술 확보

- 클라우드 스토리지를 통한 데이터 이중화, 장애복구 등 데이터 보존 신뢰성 향상 기술 확보

○ (경제적) 전자기록물 보관 및 데이터 제공 서비스에 필요한 자원을 효율적으로 제공하여 시스템 운영비용 절감 등 경제성 제공

- 저장 공간 사용량에 따른 유연한 확장을 지원하여 향후 데이터 증가 시 필요한 자원만큼 확보 가능

- 필요한 데이터를 제공을 위한 서비스를 가상화된 환경을 통해 제공하므로 사용자 수에 따른 유연한 서비스 제공 가능

○ (사회적) 행정정보, 전자기록물 등 중요한 데이터의 장기적 보관을 위한 체계적인 방안 마련 및 활용 서비스로의 연동을 위한 기반 환경 조성

- 데이터세트 유형의 전자기록의 장기보존 전략 수립을 구체화하는 방안 마련에 활용

- 테스트베드 구축 및 시험을 통해 장기보존에 적합한 보존방식을 검증하고 향후 DB유형 전자기록물 이관 및 보존에 기여

- 필요한 데이터를 제공을 위한 클라우드 기반 연계 및 확장 가능성을 제공하여 새로운 서비스로의 발전 가능성 제공

나. 활용 방안

○ 다양한 유형의 전자기록(표준전자문서, 시청각기록물 등)에 장기보존정책 및 전략에 활용

○ 현재 행정정보시스템에서 생산되는 데이터세트에 대한 이관 · 보존 · 활용 전략 및 기술 확보

○ 디지털 컴포넌트 유형별 렌더링 방법 마련 및 에뮬레이션 전략 적용 타당성 검증을 위한 에뮬레이터 프로토타입 개발을 통한 본 디지털(born digital) 기록물을 원본 기반의 장기보존 및 활용성 고취

○ 공공기관에서 생산・관리되는 DB 유형의 전자기록에 적합한 장기보존방식을 제시하여 보존・활용 기반 마련

○ 데이터 저장 및 관리가 필요한 민간 기업, 연구소 등에 구축하여 체계적이고 안정적인 기록물 관리 서비스에 활용

다. 사업 확장 가능성

○ (체계적인 에뮬레이션 체계 구축) 본 사업에서는 안정적인 데이터 이전 체계를 수립 및 다양한 솔루션 사용을 고려한 전자기록물 이전 환경을 고려하여 다양한 상용 솔루션 및 오픈소스를 활용할 수 있도록 전체 프로세스 설계

- 일반적인 클라우드 및 가상화 솔루션, 오픈소스 클라우드 프로젝트를 통해 생성된 가상환경 기반 데이터 이전 고려

- P2V 상용 솔루션 및 무료 도구 등 다양한 소프트웨어 이용한 데이터 이전 환경 고려

구분	본 사업에서 활용한 솔루션		대체 솔루션	
클라우드 & 가상화 솔루션	상용	오픈스택잇	상용	클라우드잇
				vmware
				기타 솔루션
	무료	오픈스택	무료	기타 오픈소스 SW
데이터 이전 솔루션	상용	zconverter	상용	기타 솔루션
	무료	virt-p2v	무료	기타 오픈소스 SW

<표 19> 타 솔루션 및 소프트웨어로 대체 가능한 항목

제5장 총괄연구개발과제의 연구성과

5.1 활용성과

총괄과제명	데이터세트 유형 전자기록의 장기보존기술 연구
총괄과제책임자	양동민 / 전북대학교 기록관리학과 부교수, 문화융복합아카이빙연구소 공동연구원 / 컴퓨터공학(컴퓨터네트워크・기록정보보안)

가. 연구논문

번호	논문제목	저자명	저널명	집(권)	페이지	Impact factor	국내/국외	SCI여부
1	클라우드 컴퓨팅 기반 에뮬레이션 전략을 활용한 전자기록 장기보존 방안 연구	이봉환, **한희정**, **조철용**, **왕호성**, **양동민**	한국기록관리학회지	19권 4호	1-33	-	국내 (KCI)	X

나. 학술발표

번호	발표제목	발표형태	발표자	학회명	연월일	발표지	국내/국제
1	A Study on the Long-Term Preservation of Digital Information Resources	Poster	**한희정** **양동민**	2019 ICLIS (International Conference on Library and Information Science)	2019.07.12. ~ 2019.07.13	Taipei, National Taiwan Normal University	국제
2	A Study on Administrative Information Datasets as Evidence of Public Service		**이정은** 윤은하 김건		2019.07.12. ~ 2019.07.13		국제

다. 지적재산권

번호	출원/등록	특허명	출원(등록)인	출원(등록)국	출원(등록)번호	IPC분류

라. 정책활용

· 공공기관의 데이터세트 유형 전자기록 보존 및 활용을 위한 지침 수립 지원
· 전자기록물의 마이그레이션 전략에서의 문서보존포맷 선정 체계 및 보존 방안
· 전자기록물의 에뮬레이션 방식 보존 방안 및 절차 지원

마. 타연구/차기연구에 활용

· 국가기록원 데이터세트 유형의 마이그레이션 및 에뮬레이션 전략을 위한 기술규격 표준화 연구 및 개발에 활용

바. 언론홍보 및 대국민교육

· 특기사항 없음

사. 기타

· 특기사항 없음

5.2 활용계획

가. 데이터세트의 영구기록물관리 기관 입수 및 보존에 활용

○ 현재 전자문서 이외에 데이터세트 유형에 대한 전자기록물관리를 확대할 때 적용

나. 보존포맷 규격 또는 표준에 활용

○ 기록원의 보존포맷 확대 정책에 공통기준 및 고유기준 체계 적용

다. 안정적인 기록물 보관 및 관리가 필요한 공공기관

○ 공공기관 및 국가기관에서 보유한 중요한 전자기록물의 안정적인 보관 및 관리를 위한 데이터 장기보존 시스템으로 활용

○ 오래된 전자기록물을 저장 관리하고 활용하기 위한 시스템 구축에 활용

○ 오래된 운영체제 및 소프트웨어에서 생성된 데이터의 보관, 기록 확인을 위한 시스템에 활용

라. 데이터 기록 관리가 필요한 민간기업

○ 민간 기업의 기술 개발 문서, 계약 문서 등 다양한 전자기록물을 보관하고 관리하기 위한 전자기록물 관리 시스템에 활용

○ 4차 산업혁명 시대에 급증하는 데이터로 인해 발생한 전자기록물을 효과적으로 저장하고 관리하기 위한 저장소에 활용

○ 이직, 퇴사 등으로 소멸하는 전자기록물을 효과적으로 저장하고 추후 확인하기 위한 저장 공간으로 활용

마. 다양한 정보 저장, 제공하는 데이터 비즈니스 기업

○ 데이터 비즈니스 기업이 보유한 전자기록물 정보를 효과적으로 관리하기 위한 시스템으로 활용

○ 다양한 형태의 정보로 가공된 데이터를 저장하고 애뮬레이션하여 테스트하기 위한 환경으로 활용

○ 데이터 제공을 위한 가상 환경 구축 및 서비스 운영에 활용

마. 클라우드, 스토리지, 데이터 이전 관련 사업자

○ 다양한 기관의 클라우드 솔루션, 데이터 이전 솔류션 등을 이용한 서비스 환경에 최적화된 구성으로 데이터 이전, 전자기록물 관리 사업 협업 모델 개발

○ 전자기록물 장기보존 관련 신규 비즈니스 영역 창출에 활용

제6장 기타 중요변경사항

(참여인력 변경)

○ 큐브리드 확장 SIARD Suite 개발팀 소프트웨어 개발자를 큐브리드 DBMS 전문가로 변경

제7장 참고문헌

【국내】

국가기록원 (2008), 전자기록물 문서보존포맷 기술규격(NAK 30:2008(v1.0)), 2008.

국가기록원 (2007). 행정정보시스템 데이터세트 기록관리 방안 연구보고서.

국가기록원 (2015). 데이터세트 구조분석 및 진본성 보장 기록관리 기능모델 연구.

박병주 (2011). 데이터베이스 영구보존을 위한 디지털 아카이빙 보존포맷 및 도구 개발. 충남대학교 석사학위논문.

오세라, 박승훈, 임진희 (2018). 행정정보데이터세트 사례 조사 연구. 한국기록학회지. 109-133.

왕호성, 설문원 (2017). 행정정보데이터세트 기록의 관리방안. 한국기록관리학회지. 17(3). 23-47.

행정안전부, 한국정보화진흥원 (2018). 2018년도 범정부EA기반 공공부문 정보자원 현황 통계 보고서.

【국외】

Abrams, Stephen et al, (2005). "PDF-A: The Development of a Digital Preservation Standard.", Paper presented at the 69th Annual Meeting for the Society of American Archivists, New Orleans, Louisiana, August 14 - 21.

Adams (2008). "The National Archives. Digital Preservation Guidance Note: Selecting File Formats for Long-Term Preservation.".

Barnes, Ian. (2006). "Preservation of Word Processing Documents." . *Australian Partnership for sustainable repositories*.

CENDI Digital Preservation Task Group. (2007). "Formats for Digital Preservation: A Review of Alternatives and Issues".

Clausen, Lars R. (2004). "Handling File Formats". *Statsbiblioteket*.

Eun G.Park & Sam Oh. (2012). "Examining Attributes of Open Standard File Formats for Long-term Preservation and Open Access.", *Information Technology and Libraries.* 31(4), 46-67.

Folk, Mike, and Bruce Barkstrom. (2003). "Attributes of File Formats for Long-Term Preservation of Scientific and Engineering Data in Digital Libraries.", Paper

presented at the Joint Conference on Digital Libraries, Houston, TX, May 27 - 31.

ECMA. (2008). "Office Open XML File Formats－Part 1." 2nd ed.

Hodge, Gail and Nikkia Anderson. (2007). "Formats for Digital Preservation: A Review of Alternatives and Issues.", *Information Services & Use* 27: 45 - 63.

InterPARES. (2006). General Study 11 Final Report: Selecting Digital File Formats for Long-Term Preservation, InterPARES 2 Project.

Johnson, Amy Helen. (1999). "XML Xtends its Reach: XML Finds Favor in Many IT Shops, but It's Still Not Right for Everyone.", *Computerworld*, 33(42): 76 - 81.

Knight, Gareth. (2008). Framework for the definition of significant properties. The National Archives, InSPECT Project Document.

Lesk, Michael E. (1995). "Preserving Digital Objects: Recurrent Needs and Challenges.", *In Proceedings of the 2nd NPO Conference on Multimedia Preservation.* Brisbane, Australia.

Lindley, andrew. (2013). "Database Preservation Evaluation Report - SIARD vs. CHRONOS - Preserving complex structures as databases through a record centric approach?". *Conference: International Conference on Preservation of Digital Objects (iPres)*, At Lisbon.

Malcolm Todd. (2009). "File formats for preservation.", DPC Technology Watch Series Report 09-02.

Markus Hamm, Christoph Becker. (2011). "Report on decision foctors and their influence on planning.", Scalable Preservation Environment.

Mette van Essen, Maurice de Rooij, Bill Roberts, Maurice van den Dobbelsteen (2011). Database Preservation Case Study: Review. National Archives of the Netherlands.

Müller, Eva et al. (2003). "Using XML for Long-Term Preservation: Experiences from the DiVA Project.", In Proceedings of the Sixth International Symposium on Electronic Theses and Dissertations. Berlin, May: 109 - 116.

Puglia, Steven, Jeffrey Reed, and Erin Rhodes. (2004). "Technical Guidelines for Digitizing Archival Materials for Electronic Access: Creation of Production Master Files－Raster Images.", US National Archives and Records Administration.

Rog, Judith, and Caroline van Wijk. (2008). "Evaluating File Formats for Long-term Preservation.", National Library of the Netherlands.

Sullivan, Susan J. (2006). “An Archival/Records Management Perspective on PDF/A.”, Records Management Journal 16(1): 51 - 56.

TNA(The National Archives). (2008). Selecting File Formats for Long-Term Preservation.

Ricardo Andre Pereira Freitas. (2011). Siginificant Properties in the Preservation of Relational Database.

Wilson, Andrew. (2007). Significant Properties Report. InSPECT : Significant Properties Report.

van Wijk, Caroline, and Judith Rog. (2007). “Evaluating File Formats for Long-Term Preservation.”, Presentation at International Conference on Digital Preservation, Beijing, China, Oct 11 - 12, 2007.

【인터넷자료】

The National Arvhives(TNA), <http://www.significantproperties.org.uk/>, 2019.08.03., 확인.

Arms, Caroline R. and Carl Fleischhauer. , “Sustainability of Digital Formats: Planning for Library of Congress Collections.”, <https://www.loc.gov/preservation/digital/formats/index.shtml>, 2019.08.09., 확인.

Frey, Franziska, “5. File Formats for Digital Masters.”, In Guides to Quality in Visual Resource Imaging, Research Libraries Group and Digital Library Federation., <https://pdfs.semanticscholar.org/d63d/d3c6515fafb5fdf4925a26cac2e799436a0d.pdf>, 2019.08.09., 확인.

LAC(Library and Archives Canada), Guidelines on File Formats for Transferring Information Resources of Enduring Value, <http://www.bac-lac.gc.ca/eng/services/government-information-resources/guidelines/Pages/guidelines-file-formats-transferring-information-resources-enduring-value.aspx>, 2019.08.09., 확인.

LOC(Library of Congress), Digital Preservation at the Library of Congress - Sustainability of Digital Formats: Planning for Library of Congress Collections, <http://www.loc.gov/preservation/digital/formats/index.html>, 2019.08.09., 확인.

MSA(Minnesota State Archives, Minnesota Historical Society), Electronic Records Management Guidelines File Formats, <http://www.mnhs.org/preserve/records/electronicrecords/erfformats.php>,

2019.08.09., 확인.

NARA(National Archives), Federal Records Management - Frequently asked questions about Selecting Sustainable Formats for Electronic Records, <http://www.archives.gov/records-mgmt/initiatives/sustainable-faq.html>, 2019.08.09., 확인.

WHS(Wisconsin Historical Society), Best Practices for the Selection of Electronic File Formats, <https://www.wisconsinhistory.org/Records/Article/CS15427>, 2019.08.09. 확인.

제8장 첨부 및 별첨 서류 목록

가. 첨부 서류 목록

[첨부01] 2019 ICLIS 대만학회 발표자료 1

[첨부02] 2019 ICLIS 대만학회 발표자료 2

[첨부03] 한국기록관리학 회 논문지 19권 4호 2019년 11월 게재(KCI)

[첨부04] 용어정리

나. 별첨 서류 목록

[별첨01] SIARD 2.1 표준 번역본

[별첨02] SIARD 2.1 표준 규격 분석 자료

[별첨03] SIARD Suite 빌드 및 SiardGui 실행 방법

제1세부연구개발과제 연구결과

데이터세트 유형 전자기록 보존포맷 선정 및 테스트베드 구축·시험·검증

양 동 민

전북대학교 산학협력단

제1장 제1세부연구개발과제의 최종 연구개발 목표

1. 제1세부연구개발과제의 목표

1.1 연구배경 및 목적

○ (행정정보데이터세트 기록관리 필요) 철건 구조의 표준전자문서 중심으로 설계되어 있는 현재의 기록관리 시스템의 행정정보데이터세트까지 기록관리 범위 확장 검토 필요

- 국가기록원의 기록관리 시스템들은 표준전자문서를 기반으로 수행되는 중앙정부 및 지방부처의 행정업무에 최적화되어 있음
- 철건 구조로 이루어져 있는 표준전자문서에 최적화되어 있기 때문에 다른 전자 파일(데이터세트, 시청각기록물 등)기록관리의 확장성을 위한 연구 필요
- 실시간으로 엄청나게 생산되는 행정정보데이터세트에 대한 방안 마련 시급

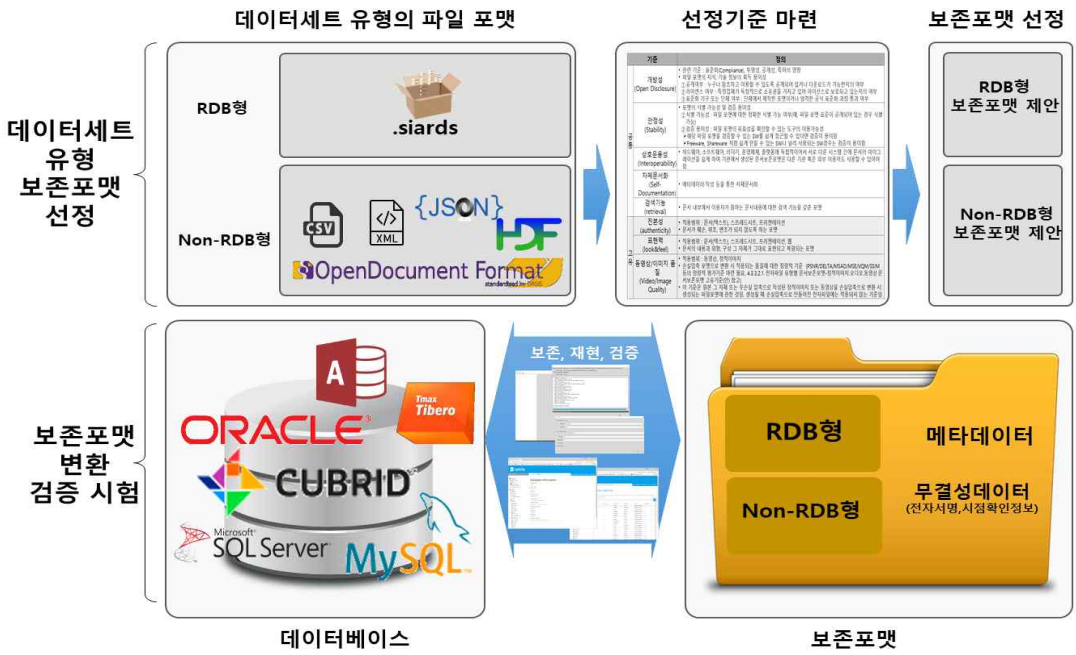
○ (데이터세트에 최적화된 문서보존포맷 선정 기준 마련 및 선정 필요) 표준전자문서에 최적화 되어 있는 현재의 PDF/A는 이 외 데이터세트에 적합한 문서보존포맷 선정 필요

- 데이터세트는 DBMS(DataBase Management System)에 저장되는 'RDB형'과 XML, CSV 또는 JSON 등의 텍스트 파일로 보관되는 'Non-RDB형'으로 분류
- 일반적으로 공개표준으로서 라이센스 없이 사용되는 Non-RDB형과 달리, DBMS는 대부분 상용화된 SW로써 기업에 대한 높은 의존성으로 인해 장기보존에 적합하지 않아 RDB형 데이터세트는 다양한 DBMS와 호환가능한 공개표준의 문서보존포맷이 필요
- 데이터세트 문서보존포맷 선정을 위해서는 선정기준체계의 사전 정립이 필요
- 다양한 유형의 DBMS의 데이터세트를 수용하기 위해서는 데이터세트 문서보존포맷의 선정기준 및 프로세스 등 선정체계에 대한 구체화가 필요하며, 선정기준은 PDF/A-1을 선정할 때의 기준 및 각국의 국립 아카이브의 문서보존포맷 선정기준을 참고하되 정량적으로 평가 가능 필수

○ (데이터세트에 대한 보존포맷 변환·재현 검증 필요) 선정기준을 통해 도출된 데이터세트의 문서보존포맷에 대한 객관적이고 실증적인 확인 필요

- 실제 테스트베드에서 다양한 DBMS에 대해서 선정된 데이터세트 문서보존포맷에 대한 다양한 검증 작업이 필요
- 각 DBMS가 제공하는 모든 데이터 타입에 대한 'DBMS → 문서보존포맷'과 '문서보존포맷 → DBMS' 실효성 검증, 대용량의 데이터세트가 안정적으로 데이터세트 문서보존포맷으로 변환·재현 실효성 검증 필요

1.2 연구의 목표



<그림 17> 제1세부연구개발과제 최종 연구 목표

○ 데이터세트 유형 전자기록 현황 및 장기보존기술 조사 및 분석

- 데이터세트 유형 전자기록의 마이그레이션/에뮬레이션 기반 장기보존기술 조사 및 분석
- 공공기관 행정정보시스템에서 데이터세트 형태・운영・관리 현황 조사 및 분석

○ 데이터세트 유형 전자기록 보존포맷 선정

- 데이터세트 유형 전자기록의 보존포맷 선정기준 수립
- 데이터세트 유형 전자기록의 보존포맷 선정기준에 따른 포맷 제시

○ 다양한 DBMS 대상으로 데이터세트 유형 전자기록 보존포맷 변환 검증 시험

- DBMS들을 대상으로 문서보존포맷 변환 검증을 위한 테스트베드 구축
- 테스트베드 기반 보존포맷 변환 검증 시험

○ 국산 DBMS 대상 변환 기능 개발

- 국산 DBMS를 대상으로 선정 및 보존포맷 변환 기능 개발

2. 제1세부연구개발과제의 목표달성도

○ 본 연구팀은 당초 계획했던 연구 목표를 모두 달성하였음

연구개발 추진내용			연구개발 일정							달성도
			5	6	7	8	9	10	11	
제1세부과제	마이그레이션	**(조사팀) 데이터세트유형 전자기록현황 및 장기보존기술 조사**								100%
		· 데이터세트전자기록마이그레이션 장기보존기술								100%
		· 공공기관 행정정보시스템 형태・운영・관리현황								100%
		(기준팀) 데이터세트유형 전자기록 보존포맷 선정기준, 평가체계 수립 및 보존포맷 선정								100%
		· 데이터세트유형 전자기록특성 도출 및 보존포맷선정기준 수립								100%
		· 보존포맷선정기준에 따른 평가체계 개발 및 보존포맷제시								100%
		(검증팀) 데이터세트전자기록보존포맷변환 검증 시험 (총 4종)								100%
		· 다양한 DBMS 테스트베드 구축								100%
		· 테스트베드 기반 보존포맷변환 검증 시험								100%
		(개발팀) 국산 DBMS 대상 변환 기능 개발								100%
		· 오픈소스 기반 국산 DBMS의 보존포맷변환 SW 개발								100%
		(총괄팀) 테스트베드 구축·시험 결과에 따른 보존적합방식 제안								100%
		· RDB와 Non-RDB에 따른 보존포맷제안								100%
		· 데이터세트유형 전자기록보존·활용을 위한 지침								100%
		(총괄팀) 사업관리								100%
		· 월간업무보고								100%
		· 보고서, 발표자료 등 작성								100%

3. 국내·외 기술개발 현황

가. 호주(NAA)

○ 호주의 NAA(National Archives of Australia)는 2018년 1월에 디지털 보존정책(Digital Preservation Policy)을 발표함

- NAA는 일반적인 문서 및 이미지뿐만 아니라 새로운 포맷인 전자메일, 시청각 기록, 혼합된 미디어(웹사이트 등), 구조화된 데이터 세트 등 다양한 전자기록물 유형을 선호·허용·위험 포맷으로 구분하여 보존

- NAA는 기본적으로 마이그레이션 전략을 사용하지만, 원본 비트스트림을 보존하고 모든 버전을 보유하여 지속적인 접근성을 보장

- NAA은 장기보존을 위해서 Manifest Maker, XENA(Xml Electronic Normalising for Archives), DPR(Digital Preservation Recorder), Checksum Checker 개발

- NAA는 Manifest Maker를 통해 체크섬을 생성하고, Checksum Checker를 통해 검증하며, DPR을 사용하여 기록의 바이러스, 무결성을 체크하고 보존포맷(XENA)으로 변환(NAA, 2011)

나. 캐나다(LAC)

○ LAC(Library and Archives Canada)는 2017년 11월 디지털 보존프로그램을 위한 전략(Strategy for Digital Preservation Program) 개발

- LAC가 개발한 디지털 보존프로그램 전략은 국제 표준인 ISO 14721:2012 OAIS 참조 모형을 통하여 디지털 아카이브의 기능과 역할을 설명하고 디지털 보존프로그램을 구현하는데 필요한 핵심 요소를 정의

- LAC의 장기보존 정책에서는 디지털 보존프로그램을 개발하기 위한 로드맵을 설계하는데 중점을 둠

- LAC는 장기보존 전략 중 마이그레이션 전략을 채택하고 있으며, 보존포맷 요건은 다음과 같이 정의(LAC, 2017)

1) 공개성/투명성

· 파일포맷 지식과 기술정보 축적이 용이

2) 보존표준으로 채택

· 국가도서관, 아카이브 등에서 공식으로 채택

3) 안정성/호환성

· 이전/이후 기종과 호환

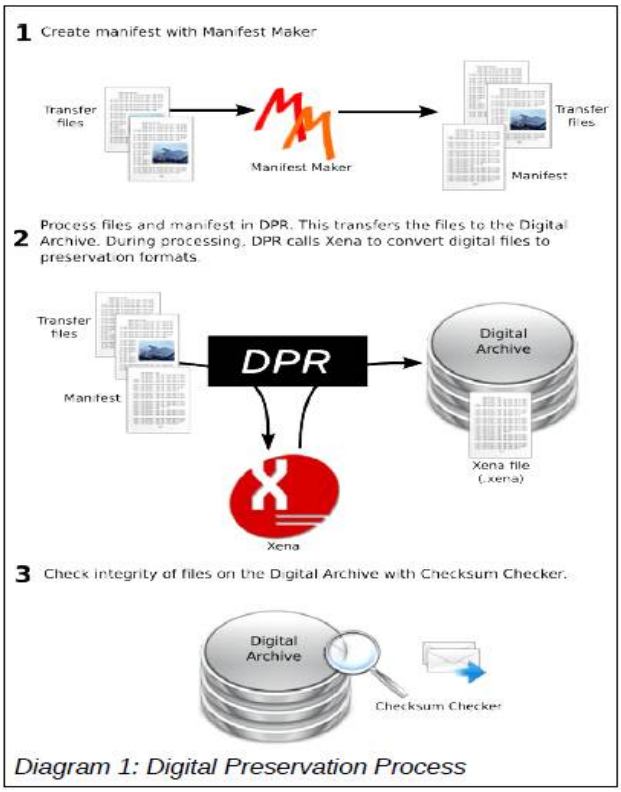
· 파일 변형으로부터 보호되는 정도

· 시간 경과에 따른 포맷 업데이트 또는 대체 버전의 상대적 빈도

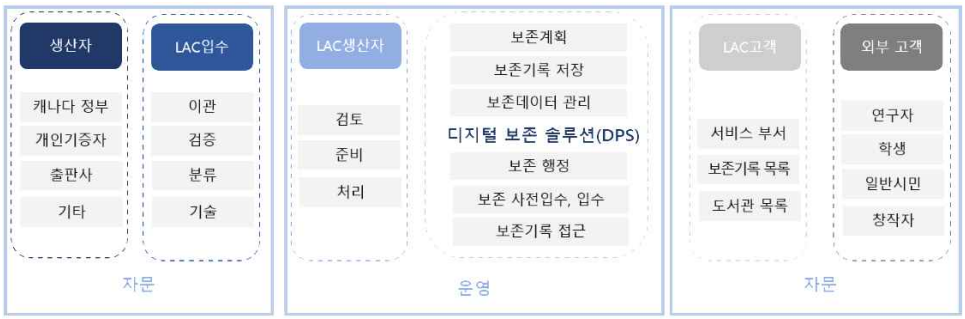
4) 의존성/상호운용성

· 포맷이 특정한 하드웨어 또는 소프트웨어에 의존하는 정도

· LAC는 포맷 마이그레이션 정책을 통해 구형화 위험 포맷 및 매체에 대한 선호 포맷 가이드라인 제시



<그림 18> Digital Preservation Process (NAA, 2011)



<그림 19> LAC의 디지털 보존프로그램 (LAC, 2017)

다. 영국(TNA)

○ 2011년 영국 TNA(The National Archives) '디지털 보존 정책아카이브즈를 위한 지침'(Digital Preservation Policies: Guidance for archives)'

○ 2017년 '디지털 전략 2017-2019'(Digital Strategy2017-2019)에서 보완

- 현재 TNA는 문서, 이미지, 이메일, 비디오, 웹사이트 등에 집중하고 있으나 구조화된 데이터세트와 컴퓨터 코드까지 보존 범위를 확대하기 위한 기술 개발을 고려하고 있음
- TNA는 장기보존 전략으로서 원본 비트를 유지하는 에뮬레이션 전략 선호
- TNA는 보존도구로서 DROID, PRONOM, COPTR 개발

기준	개념	특징	환경
DROID	포맷 식별 도구	- 내부 서명을 이용하여 식별 - 오픈소스, 전 세계적으로 널리 사용 - 현재 6.4v - 버전, 나이 및 크기, 마지막으로 변경된 시기를 알 수 있음 - csv파일로 내보내기 가능	JAVA 1.7 또는 1.8 Standard Edition (SE) 환경
PRONOM	포맷 정보 레지스트리	- 정부 부서에 레코드를 저장할 때 사용되는 파일 포맷 정보를 관리 - 1,300개 이상 개별 파일 형식 항목을 포함 - 현재 PRONOM 6.2v - 마이그레이션 전략 시행 시 중요 정보 제공	Window 2000
COPTR	보존 도구 레지스트리	- 보존 전문가가 특정 보존 작업을 수행하는 데 필요한 도구를 찾고, 평가하는 도구 - 현재 445가지 도구 보유 - OAIS 참조모델과 DCC의 디지털 큐레이션 생애주기 모델 참조	x

<표 20> TNA 디지털 보존 도구

라. 미국(NARA)

○ 미국 NARA(National Archives)은 2017년 8월 디지털기록 보존전략(Strategy for Preserving Digital Archival Materials)을 발표함

- 디지털기록 보존을 위한 6가지 전략 제시

1) 표준과 절차의 문서화(Documentation of Standards and Procedures)

2) 우선순위(Prioritization)

3) 파일 관리(File Management)

4) 진본성(Autnenticity)

5) 보존 메타데이터(Preservation Metadata)

6) 조직적인 관계(Organizational Relationships)

- NARA에서는 보존활동을 디지털 보존 인프라 기반 구조, 데이터 무결성, 포맷 및 매체 지속성, 정보보안으로 구분하여 제시

- NARA는 다양한 유형의 전자기록물을 보존하기 위해 마이그레이션 전략을 채택하고 포맷의 경우 선호 · 허용 · 위험 포맷으로 구분하여 체계적으로 관리

마. 스위스(SFA)

○ SFA(Swiss Federal Archives)의 장기보존 정책은 2009년에 발표된 'digital archiving policy'와 2015년에 발표된 'Federal Archives Strategy 2016 - 2020'에서 볼 수 있음

- SFA는 텍스트 문서에서 사진, 녹음, 매우 복잡한 데이터베이스에 이르기까지 다양한 형태를 보존

- SFA는 보존 전략으로 3가지 전략을 제시

가) 마이그레이션 원칙

· SFA는 마이그레이션 전략을 사용하여 전자기록물을 필요에 따라 변화하는 환경에 맞추어 선정된 보존포맷으로 변환하여 보존

· 무손실 변환을 강조하며, 모든 변환은 변경 사항이 기록되며, 문서는 어떤 경우에도 복원 가능하도록 함

· 에뮬레이션 전략을 사용하지 않으며, 원래의 하드웨어와 소프트웨어를 보존하지 않음

나) 애플리케이션과 분리

· SFA는 특정 IT환경(응용프로그램, 데이터베이스, 운영체제, 하드웨어)에서 데이터를 분리하는 전략을 추구

· 응용프로그램은 보존하지 않음

· 특정 응용프로그램(예. 데이터베이스)은 기록관리 업무 시 검증하고 데이터를 사용할

필요가 있을 때 보존 할 수 있음(예: 데이터모델)

다) 원본 데이터의 매체는 저장하지 않음

· 디지털 문서는 원본 데이터 매체를 고려하지 않고 보관

· 보존 프로세스가 완료될 시 원본 데이터 매체에서 데이터를 삭제하고, 이관이 완료되면 이관에 사용된 보존 매체는 폐기되거나 반환됨

바. 국가기록원

○ 국가기록원의 국가기록관리혁신추진단은 전자기록관리 체계 재설계를 위해 장기보존 정책, 전자기록 유형별 관리체계 및 포맷정책 재설계 등을 혁신과제로 추진

○ 그 중 전자기록 장기보존 정책의 경우, 전자기록 장기보존에 대한 기본 정책 없이 유형별 및 정보기술별 절차만이 존재. 이에 전자기록의 장기보존 목표, 전략 등 기본 정책을 수립하고자 함

· 세부과제 7-1 : '장기보존을 위한 기본 정책 수립'

· 세부과제 7-2 : '전자기록 유형별 관리 및 보존 설계 및 제도화'

○ 현재 혁신과제 이행과정 중 중간보고 내용(국가기록원, 2018)

- 전자기록 장기보존 정책 수립

1) 전자기록 장기보존 기본정책 수립

2) 전자기록 장기보존 상세 이행계획 수립

3) 전자기록 장기보존 정책 및 이행 점검 체계 마련

4) 전자기록 장기보존 정책과 이행 지원을 위한 연구기능 강화

- 전자기록 유형별 관리체계 및 포맷정책 재설계

1) 전자기록물 유형별 관리체계 및 포맷정책의 기본방안 수립

2) 전자기록물 유형별 관리체계 재설계

3) 전자기록물 유형별 통합 수집·관리를 위한 시스템 고도화

- 전자기록 유형별 보존포맷(안)이 검토 및 향후 검토 필요사항으로 등장

· 파일형(NEO 1.0, 2.0, VEO 1.0, 2.0, XENA), 폴더형(VEO 3.0, Bagit, DSpace)

○ 장기보존 전략의 경우, 표준전자문서의 장기보존포맷으로의 변환 외에는 부재한 것으로 분석됨

○ 전자기록물 문서보존포맷 기술규격 (NAK 30:2008(v1.0)).

- 국가기록원에서 정한 보존기간 10년 이상인 전자기록물에 대한 문서보존포맷

· 편재성, 안정성, 메타데이터지원, 상호운영성, 진본성, 표현력, 검색기능에서 우수

· 국가기록원은 현재 PDF/A 기반 표준전자문서 시스템으로 최적화 되어 있음

	XML	Text	이미지 (TIFF/BITMAP)	PDF	CSD	PDF/A
공개용 표준	상	상	상	하	하	상
편재	상	상	상	중	중	상
안정성	상	하	상	상	상	상
메타데이터 지원	상	하	하	상	상	상
상호운영성	상	상	상	상	상	상
진본성	상	상	상	상	상	상
처리능력	상	하	중	상	상	상
표현력	중	하	상	상	상	상
검색 기능	상	상	하	상	상	상

<표 21> 다양한 문서 포맷 비교표 - PDF/A

○ 전자기록물 장기보존포맷 기술규격 (NAK 31 : 2017(v2.1))

- 장기보존포맷 고려사항

· 자체충족성 : 기록물이 생산된 당시의 내용을 그대로 재현하여 읽고 이해할 수 있도록 하기 위하여 전자기록물은 시스템, 외부데이터 등에 독립적이어야 함

· 자체문서화 : 전자기록물은 자체적으로 기록물과 관련된 기술(記述)과 맥락에 대한 이해를 줄 수 있는 정보를 포함하여, 미래의 이용자들이 장기 보존된 기록물의 내용을 이해할 수 있도록 해야함

· 구조화된 텍스트 인코딩 : 캡슐화하는 기록물은 바이너리 데이터보다는 구조화된 텍스트 형식으로 인코딩되어야 함

· 무결성 : 전자기록물이 위조 또는 변조 되지 않았음을 보장하여 기록물의 법적증거를 확보하여야 하며, 무결성을 보장하기 위해 인증정보를 적용해야 함

- 장기보존포맷 구성요소

· 원문 : 생산자가 생산 또는 접수한 전자기록물 원본(진본)으로 진본성을 보장하기 위해 포함

· 문서보존포맷 : 문서 생산 또는 접수 당시의 애플리케이션이 없이도 해당 문서의 내용과 외형을 그대로 재현한 포맷으로, 시간과 기술변화에 상관없이 이용자가 문서 내용에 접근할 수 있게 함

· 장기보존포맷 메타데이터 : 기록물의 생산 접수부터 관리·보존에 이르는 전 과정을 기술(記述)한 정보로, 기록물 생애주기 전 기간에 걸쳐 진본성, 신뢰성, 무결성, 이용가능성을 보장하며, 기록물을 관리하고 보존·이해할 수 있도록 지원함

· 전자서명 : 전자기록물의 진본성 및 무결성 보장을 위해 장기보존포맷에 포함되는 정보로, 전자서명은 인증정보와 잠김인증정보로 구성되어 있음

제2장 제1세부연구개발과제의 연구개발 내용 및 방법

1 연구내용

1.1 데이터세트 유형 전자기록 현황 및 장기보존기술 조사 및 분석

○ 데이터세트 유형 전자기록 관련 마이그레이션 기반 장기보존기술 조사 및 분석

- 다양한 아카이브 기관들의 데이터세트 유형 전자기록을 위한 문서보존포맷(DB형(RDB형, Non-RDB형), Non-DB형 등) 표준 및 특징 조사 및 분석

(※ SIARD(Software Independent Archiving of Relational Databases) 2.1 등)

- 다양한 장기보존포맷(파일형, 폴더형) 표준의 특징 조사 및 분석 (NEO, VEO2/VEO3, XENA, BagIT, E-ARK AIP 등)

선호포맷(Preferred format)

구분			미국 NARA	캐나다 LAC	호주 NAA
Non-DB type	JSON		√		
	CSV		√	√	
	XLS	XLS			
		XLSX			
	ODS		√		
	TXT	ASCII	√	√	
		Unicode			
	XML		√		
	EBCDIC				
DB type	DBF				
	SIARD				

허용포맷(Acceptable Format)

구분			미국 NARA	캐나다 LAC	호주 NAA
Non-DB Type	JSON				√
	CSV				√
	XLS	XLS	√	√	√
		XLSX	√		√
	ODS			√	√
	TXT	ASCII			√
		Unicode			√
	XML				√
	EBCDIC			√	√
DB type	DBF			√	
	SIARD				√

<표 22> 해외 문서보존포맷 선정기준 현황

○ 데이터세트 유형 전자기록 관련 에뮬레이션 기반 장기보존기술 조사 및 분석

- 전가상화(full virtualization), 반가상화(para-virtualization), OS레벨 가상화(OS-level virtualization), 콘테이너(container) 기술 등의 특징을 조사 및 분석

- 아키텍처 및 OS별 지원 가능한 에뮬레이션 방법들의 특징을 조사 및 분석

(※ 아카이브 기관 및 다른 분야 현황(예, 디지털포렌식, 게임, 시스템 마이그레이션 등)

○ 공공기관의 행정정보시스템에서 생산·관리되는 데이터세트 형태·운영·관리 현황

- 데이터세트 단독, 데이터세트+응용프로그램, 데이터세트+WAS(Web Application Server)+Web Server 등 다양한 형태로 운영되는 행정정보시스템을 대상으로 함

- 다양한 DBMS 시스템들의 사용 현황 조사 및 분석(DBMS 종류, 데이터 크기, 데이터 타입 종류, 외부 파일 규모 등)

1.2 데이터세트 유형 전자기록 보존포맷 선정기준 수립 및 선정

○ 데이터세트 유형 전자기록의 보존포맷 선정기준 수립

- RDB(Relational DB)와 Non-RDB(Non-Relational DB)를 구분하여 고유기준 제시
- 국내 · 외 데이터세트 유형 전자기록의 문서보존포맷 및 선정 기준 조사 및 분석 (XML, CSV, JSON, XSL, ODF, SIARD, HDF 등)
- 데이터세트 유형의 고유한 특성 추출 및 평가 기준 도출 (※공통기준 중 반드시 준수해야 하는 필수사항과 선택사항을 구분하여 제시)

기준	캐나다 (LAC)	INTERPARES 2 Project	미국 (LOC)	영국 (TNA)	미국 (NARA)	한국 (NAK)
개방성/투명성	○	○	○	○	○	○
채택	○	○	○	○	○	○
안정성/호환성	○			○		○
종속성/상호운용성	○	○	○	○		○
표준화(규격화)	○					
압축		○			○	
특허의 영향			○	○		
기술 보호 메커니즘			○		○	
메타데이터 지원			○	○	○	○
진본성						○
표현력						○
검색기능						○

<표 23> 해외 문서보존포맷 선정기준 현황

선정기준	선정 기준 정의
개방성/투명성	파일 포맷의 지식, 기술 정보의 획득 용이성
채택	국제적으로 국립도서관 · 기록관 등의 기관이 공식적으로 채택된 정도
안정성/호환성	버전 공개빈도 주기, 호환가능성 및 변경·변질에 대한 탄력성 정도
종속성/상호운용성	포맷이 특정 하드웨어, 소프트웨어, 리더기에 의존하는 정도
표준화(규격화)	포맷이 엄격한 공식 표준화 과정을 통과하는 정도
압축	압축 및 무손실 압축 여부
특허의 영향	콘텐츠를 유지하는 아카이브 기관의 능력이 특허에 의해 금지되는 정도
기술 보호 메커니즘	콘텐츠 보존을 방해하는 암호화와 같은 메커니즘 구현
메타데이터 지원	메타데이터 작성 등을 통한 자체문서화(self-documentation)
진본성	문서가 훼손, 위조, 변조가 되지 않도록 하는 포맷
표현력	문서의 내용과 외형, 구성 그 자체가 그대로 표현되고 복원되는 포맷
검색기능	문서 내부에서 이용자가 원하는 문서내용에 대한 검색 기능을 갖춘 포맷

<표 24> 해외 문서보존포맷 기준 현황

○ 데이터세트 유형 전자기록의 보존포맷 선정기준에 따른 포맷 제시

- 선정기준에 따라 평가 지표 제시, 가중치 부여, 가중치 산출 등을 반영한 평가절차, 방법, 근거 도출

- 평가에 따른 권고 포맷을 선정하며, RDB형과 Non-RDB형 각각 제안

기준	내용	평가
개방성	• 공개여부 : 공개되어 있음 ➢ Swiss e-Government standards (eCH) : https://www.ech.ch/standards/38716 ➢ SIARD suite software : https://github.com/sfa-siard/SiardGui/releases	상
	• 라이센스 여부 : Open Source (CDDL-License) on Github	상
	• 표준화 기구 또는 단체 여부 : Swiss e-Government standard (eCH-0165)	중
안정성	• 식별 가능성 : XML, SQL:1999 기반의 ZIP 패키지로 DB구조와 데이터가 SIARD 표준에 공개되어 있음	상
	• 검증 용이성 : SIARD Suite 및 GUI 의 참조 프로그램을 통해 검증할 수 있음	상
상호운용성	• JAVA로 작성된 참조 프로그램이 있음(예 : SiardGui, SIARD Suite)	상
자체문서화	• 메타데이터 기능이 있음 ➢ 관계형 데이터베이스의 구조 메타데이터를 나타내는 스키마 포함	상
검색기능	• XML 기반의 텍스트 파일이므로 검색 가능	상

<표 25> 선정기준에 따른 문서보존포맷 평가 방안 예시 (RDB형: SIARD)

1.3 다양한 DBMS 대상으로 데이터세트 유형 전자기록 보존포맷 변환 검증 시험 및 국산 DBMS 대상 변환 기능 개발

○ 공공기관에서 주로 사용하고 있는 DBMS들을 대상으로 문서보존포맷 변환 검증을 위한 테스트베드 구축 (큐브리드, AltiBase, 오라클, MySQL, MS SQL Server 등)

○ 테스트베드 기반 보존포맷 변환 검증 시험

- 보존포맷 변환 프로세스 검증(사전준비→변환→사후처리)

- 원천 데이터세트와 보존포맷으로 재현한 데이터세트 간 정합성 검증(dataset → (변환) → 보존포맷 → (재현) → dataset)

- DBMS에서 제공하는 다양한 데이터 타입에 대한 시험(문자, 숫자, 파일 등)

- DBMS에 적재된 데이터양에 따른 보존포맷 변환 검증 시험(5GB, 10GB, 50GB, 100GB 등)

- RDB의 경우, 주요 DBMS별 DB구조, 저장프로시저, 트리거 등의 요소별로 확인하여 원천 데이터세트의 보존포맷 변환 후 진본성(authenticity) 유지여부 검증

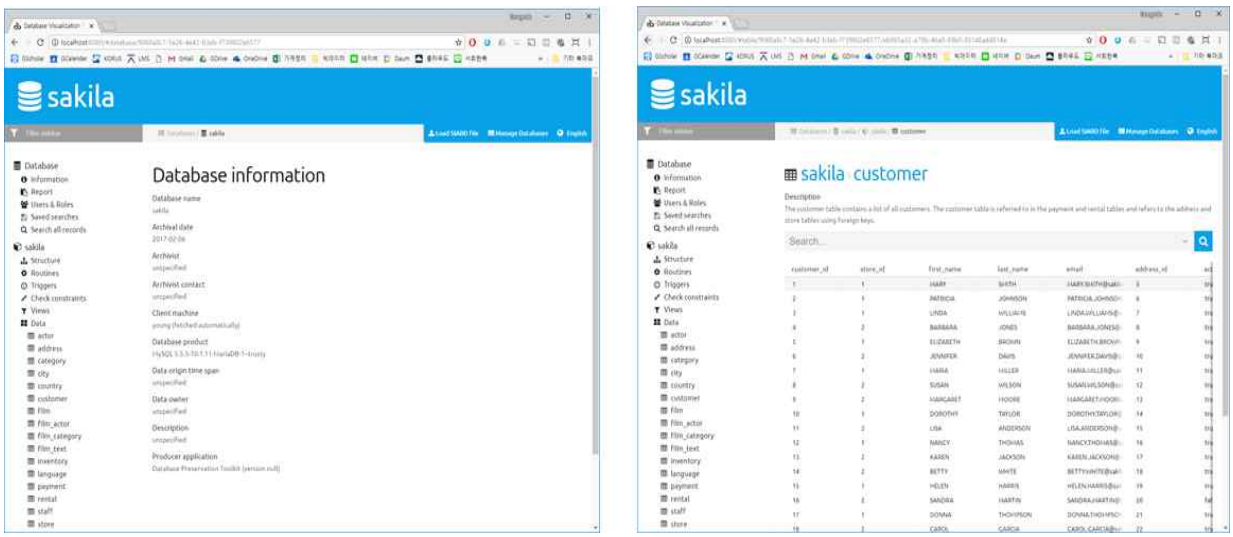
- 변환 검증 시험을 통해 도출된 문제점 조사·분석 및 해결책 제시

◦ DBMS 외부 파일시스템에 저장되는 파일 처리 과정 분석 및 방안 제시

<그림 20> 데이터세트 유형 전자기록 보존포맷 변환 검증

◦ 공공기관에서 많이 사용하는 국산 DBMS를 대상으로 보존포맷 변환 기능 개발

- 국산 DBMS에서 데이터세트를 추출하여 보존포맷으로 변환하는 기능
- 보존포맷에서 데이터를 추출하고 조회(재현)하는 기능
- 보존포맷으로 변환된 데이터세트의 정합성, 무결성 검증 기능
- 보존포맷에 저장된 데이터세트를 보존 DBMS에 적재하는 기능



<그림 21> 보존포맷 시각화 기능

1.4 테스트베드 구축·시험 결과에 따른 보존적합방식 제안

○ RDB와 Non-RDB에 따른 보존포맷 제안

○ RDB의 경우, 시스템・SW 환경, 시스템・SW 연계상황, 규모 등에 따라 보존방안 제안

○ 공공기관의 데이터세트 유형 전자기록 보존 및 활용을 위한 지침 작성 및 제출

○ 검증과정 및 국산 DBMS 대상으로 보존포맷 변환 SW 개발 시 발생하는 문제점과 장·단점 등을 조사·분석하여 이를 반영한 보존포맷(안) 제시

2. 연구방법

2.1 데이터세트 유형 전자기록 현황 및 장기보존기술 조사

○ 국내・외 대표 국립 아카이브 선정(미국:NARA, 영국:TNA, 캐나다:LAC, 호주:NAA 등) 및 장기보존기술 현황 조사

○ 국립 아카이브의 장기보존기술 비교 분석 및 시사점 도출

- 문서보존포맷으로 선정한 파일포맷들의 특징 및 선정기준 비교 분석 및 시사점 도출
- 장기보존포맷으로 선정된 파일포맷의 구조(원문, 무결성데이터, 메타데이터 등)의 특징을 비교 분석한 후 시사점 도출
- 에뮬레이션을 전략으로 삼고 있는 TNA(The National Archives)의 현황 조사 및 분석
- 에뮬레이션의 시스템 관점에서의 특징・한계점 분석 및 시사점 도출

○ 공공기관 행정정보시스템 현황 자료 확보 및 인터뷰 면담 실시

○ 공공기관 행정정보시스템의 생산・관리되는 데이터세트 형태・운영・관리 현황

- 행정정보시스템의 서버 및 정보자원 구성 조사 및 분석
- 생성되는 데이터베이스 종류・구조・크기・데이터타입 등의 현황 조사 및 분석

2.2 데이터세트 유형 전자기록 보존포맷 선정

○ 데이터세트 유형 전자기록의 보존포맷 선정기준 수립

- 데이터세트를 표현하는 다양한 파일포맷 표준 조사 및 분석(XML, CSV, JSON, XSL, ODF, SIARD, HDF 등)
- 국외 대표 국립 아카이브 선정(미국:NARA, 영국:TNA, 캐나다:LAC, 호주:NAA 등) 및 보존포맷 선정 기준 조사 및 비교 분석

- 데이터세트 전자기록의 고유 특징을 반영하고 진본성 및 무결성을 보장할 수 있는 고유 특징 도출 및 선정 기준 정의

- 선정 기준에 따라 평가 지표 제시, 가중치 부여, 가중치 산출 등을 반영한 평가절차, 방법, 근거 도출

○ 데이터세트 유형 전자기록의 보존포맷 선정 기준에 따른 보존포맷 제시

- 도출된 선정 기준에 따라 다양한 파일포맷들을 적용하여 RDB형과 Non-RDB형을 위한 각각의 보존포맷 제안

2.3 다양한 DBMS 대상으로 데이터세트 유형 전자기록 보존포맷 변환 검증 시험 및 국산 DBMS 대상 변환 기능 개발

○ 다양한 DBMS 대상으로 데이터세트 유형 전자기록 보존포맷 변환 검증 시험

- 발주기관과 협상하여 분석 대상 DBMS 선정 및 테스트베드 구축

- 오픈소스 프로젝트 기반 원본 데이터세트 및 보존포맷 사이 변환 SW 개발

- 정합성 검증 원칙 설정: (dataset → (변환) → 보존포맷 → (재현) → dataset')의 변환 및 복원과정 후, dataset=dataset'여부 조사

- DBMS 지원 데이터 타입 정합성 검증 및 적재 데이터량에 따른 정합성 검증 시나리오 작성 및 정합성 검증

- RDB의 경우, 주요 DBMS별 DB구조, 저장프로시저, 트리거 등의 요소별로 확인하여 원천 데이터세트의 보존포맷 변환 후 진본성(authenticity) 유지여부 검증

○ 국산 DBMS 대상 변환 기능 개발

- 발주기관과 협의하여 국산 DBMS 선정

- 오픈소스 프로젝트 기반 원본 데이터세트 및 보존포맷 사이 변환 SW 개발

- 데이터세트를 추출하여 보존포맷으로 변환하는 기능, 보존포맷에서 데이터를 추출하고 조회(재현)하는 기능, 보존포맷으로 변환된 데이터세트의 정합성, 무결성 검증 기능, 보존포맷에 저장된 데이터세트를 보존 DBMS에 적재하는 기능 개발 및 테스트

2.4 테스트베드 구축·시험 결과에 따른 보존적합방식 제안

○ 데이터세트 유형, 시스템 구성 등에 따라 마이그레이션 또는 에뮬레이션 방안 제안

○ 마이그레이션 방안 적용 시, 선정된 데이터세트 보존포맷으로 변환·보존·검증·활용을 위한 지침 작성, 에뮬레이션 방안 적용 시, 실험된 클라우드 기반 에뮬레이션 방식으로 변환·보존·검증·활용을 위한 지침 작성

제3장 제1세부연구개발과제의 최종 연구개발 결과

1. 데이터세트 유형 전자기록 현황과 장기보존기술 조사

- 데이터세트 유형의 장기보존기술 중 현재 가장 많이 논의되고 있는 SIARD 2.1 표준에 대해 조사함
- SIARD 2.1 표준은 메타데이터와 테이블 데이터가 결합된 구조로 원본 DBMS를 사용할 수 없게 되더라도 데이터세트의 내용을 확인할 수 있는, 독립적인 단일 파일 생성이 가능한 표준임
- SIARD 2.1 표준을 따르는 SIARD Suite은 오픈소스 프로젝트이기 때문에 오픈소스 프로젝트 라이선스 정책에 대한 조사도 병행하였으며, 각 라이선스 정책 별로 공개 범위가 상이하나 공익적 차원에서 수정하여 재배포하는 경우에도 이를 공개하는 것이 바람직하다고 판단 됨
- SIARD 표준은 스위스, 덴마크, 포르투갈을 중심으로 개발, 활용되고 있음

1.1 SIARD 2.1 표준

- SIARD 2.1 표준은 관계형 데이터베이스에 저장된 데이터세트를 독립적인 단일 파일로 장기보존 할 수 있도록 개발된 표준임. 이는 메타데이터와 테이블 데이터가 결합된 구조로, XML과 SQL:2008 표준에 기반하여 데이터세트로의 접근과 교환을 가능하게 하였음
- SIARD 표준은 관계형 데이터베이스의 데이터세트 장기보존 방법 중 현재 유일한 오픈소스 표준이기 때문에, 주관기관과의 협의 하에 선제적으로 현황 조사를 하였음

가. SIARD 2.1 표준 개요

○ SIARD(Software Independent Archival of Relational Databases)는 관계형 데이터베이스에 저장되어 있는 데이터세트를 소프트웨어와 독립적으로 하나의 파일로 '장기보존'할 수 있도록 개발된 표준임

○ 원본의 데이터베이스 소프트웨어를 사용할 수 없게 되더라도 XML과 SQL:2008 표준에 기반하여, 데이터베이스 데이터의 접근과 교환을 가능케 할 수 있음
(※ 장기보존: 비트스트림을 유지하여 영구적으로 정보를 보존하고, 이용자가 읽고 이해할 수 있도록 데이터를 해석하고 표현하는 능력을 일컬음)

○ SIARD의 첫 번째 버전 SIARD 1.0은 2007년 SFA(Swiss Federal Archive: 스위스 연방기록원)에서 개발되어 2013년에 eCH-0165라는 표준으로 제정됨

○ 2016년에는 유럽 E-ARK 프로젝트의 일환으로서 SIARD의 버전 2.0이 나왔으며, 현재 SIARD 2.1까지 제시되었음. <그림 22>과 같음

<그림 22> 이전 SIARD 표준의 변천

○ SIARD는 Unicode, XML, SQL:2008, URI(Uniform Resource Identifier), ZIP 등 표준 기반

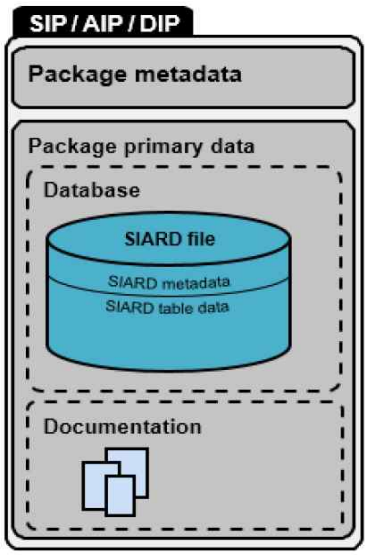
○ SIARD 2.0이 개발되면서 SIARD 1.0에 비해 추가된 새로운 기능은 아래와 같음

- SQL:1999에서 SQL:2008로 업그레이드되면서 SQL:2008의 모든 데이터 타입을 지원하며 사용자 정의 데이터 타입(UDT: User-Defined Data Type)도 사용 가능

- 정규표현식(Regular Expression)을 사용하여 데이터 타입 규칙 준수 여부 검증 가능

- SIARD파일이 데이터베이스에 속하지만 외부에 저장되어 있는 대용량의 객체를 "file:" URI를 이용하여 참조할 수 있음. 마지막으로 압축 방법으로 deflate 방식을 지원함

○ OAIS 모델 패키지 구조와 독립적으로 설계됨. OAIS 패키지 메타데이터와 관계없이 자체 메타데이터를 가지고 있으며, 다른 문서들(외부 LOB파일, 외부 파일 이름에 대한 변환 맵, DB 문서, DB 구조와 관련 문서 등)과 함께 보존되는 것으로 가정함. <그림 23>과 같음



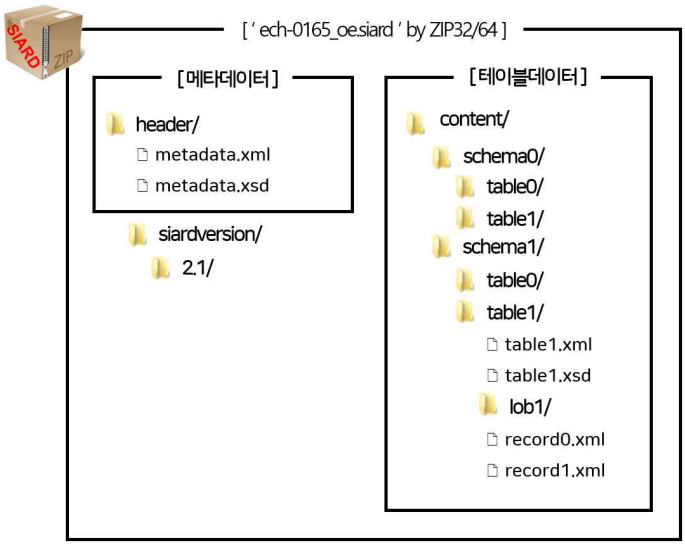
<그림 23> SIARD 정보 패키지

○ 각 구성요소에 대한 세부 요구사항은 SIARD 2.1 포맷 표준 번역문 참조([별첨01] 참고)

나. SIARD 2.1 표준 규격 분석

○ SIARD 아카이브 구조: 메타데이터와 테이블데이터가 결합된 구조. 하나의 관계형 DB는 단일의 SIARD 파일로 저장됨

○ 모든 DB 콘텐츠는 XML 스키마 1.0의 스키마 정의에 따라, XML 1.0 포맷의 파일 집합으로 보관됨. 스키마 정의와 SQL코드는 SQL:2008을 따르며, <그림 24>와 같음



<그림 24> SIARD 아카이브 내부 파일구조의 개요도(예시)

○ metadata.xml과 metadata.xsd

- metadata.xml: content/ 폴더 내의 구조, 즉 스키마 및 테이블의 개수에 대한 값을 가짐. 또한 지정된 각 테이블의 열 개수에 대한 값을 가짐. <그림 25>와 같음

- metadata.xml에 대한 스키마 정의파일. 이 파일에서는 모든 레벨의 메타데이터를 지정하여, meatadata.xml이 지정된 메타데이터를 저장하도록 함

○ table[number].xml과 table[number].xsd

- table[number].xml: XML 스키마 정의에 근거하여 원본 DBMS의 테이블 데이터를 저장한 XML 파일. 각 데이터에 대한 메타데이터는 meatadata.xsd에서, 테이블의 각 열에 대한 데이터 타입의 유효성에 대한 값은 table[number].xsd에서 가짐

- table[number].xsd: 각 테이블의 테이블 데이터에 대한 XML 저장 포맷을 나타내기 위해 생성된, 스키마 정의파일. 각 table[number].xml 파일은 table[number].xsd에 의한 유효성 검사를 통과하여야만 함

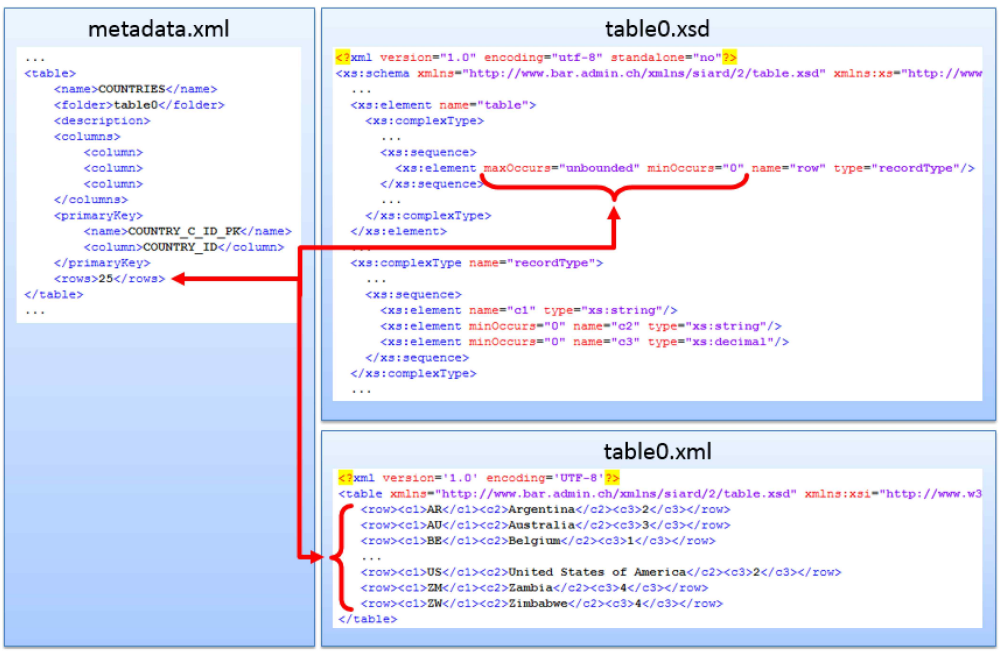
<그림 25> metadata.xml과 SIARD 파일 내 폴더 구조의 대응(예시)

○ metadata.xml과 table[number].xsd

- metadata.xml에 기재된 메타데이터 정보는 각 테이블의 스키마 정의 파일(table[number].xsd)에 기재된 내용과 같아야 함

- 대상: 각 테이블의 행과 열 개수, 열 정의에 대한 데이터 타입 정보, 열의 순서, 테이블 정의에 있는 필드 시퀀스. <그림 26>와 같음

○ 이외 SIARD 표준 규격에 대한 분석은 SIARD 2.1 표준 규격 분석 자료 참조([별첨02])



<그림 26> metadata.xml과 table0.xsd 및 table0.xml의 table0에 대한 행 개수 대응(예시)

1.2 SIARD 오픈소스 프로젝트 라이선스 정책

- SIARD는 오픈소스 프로젝트이므로 각 관련도구의 소스코드 활용에 앞서, 라이선스 정책의 분석이 필요함. 이에 따라 대표적인 SIARD 관련 도구들을 중심으로, GPL/LGPL/CDDL 라이선스 정책 간의 차이점을 기술하였음
- SIARD Suite과 DB Preservation Toolkit 각각 CDDL과 LGPL이며, 자체 개발한 코드를 별도의 파일로 작성하면 공개 의무 없음

○ 라이선스 정책 분석 대상: SIARD Suite, DB preservation toolkit

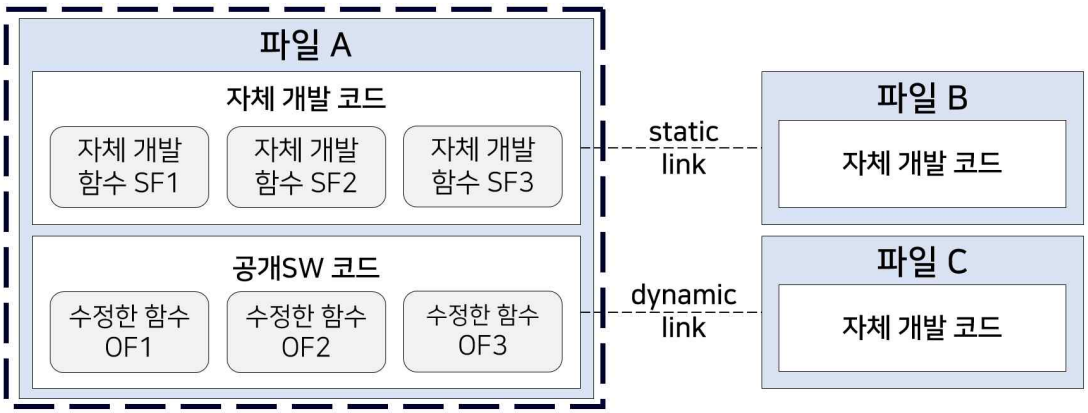
- 관련 도구의 소스코드 활용을 염두에 둔 라이선스 정책 분석. <표 26>과 같음

도구	관리 주체	주요 내용
SIARD Suite	스위스연방기록원(SFA)	· 지원 포맷: SIARD 2.1 · 오픈소스 라이선스: CDDL v1.0
DB preservation toolkit	KEEP SOLUTIONS	· 지원 포맷: SIARD 1, SIARD 2, SIARD DK · 오픈소스 라이선스: LGPL v3.0

<표 26> SIARD 관련 도구의 주요내용

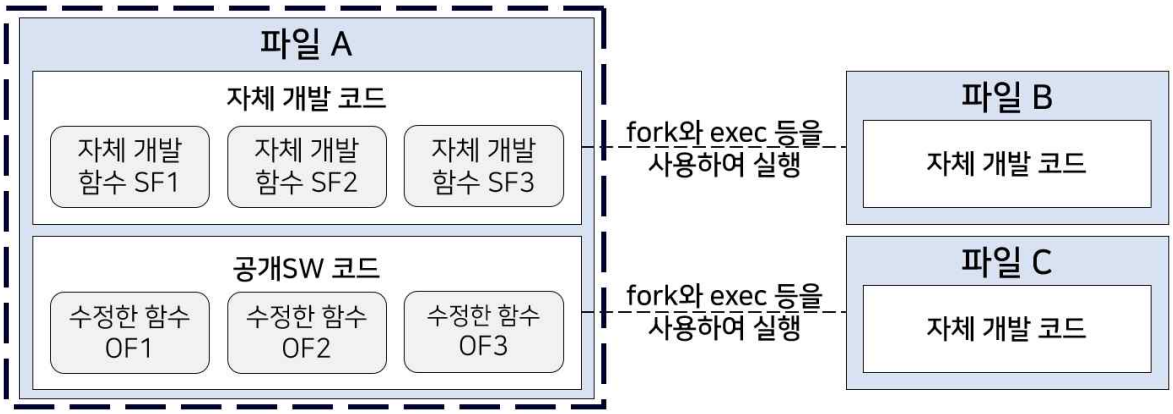
○ 오픈소스 라이선스 비교 (GPL v3.0, LGPL v3.0, CDDL v1.0)

- GPL 공개SW 사용 시 링크에 관계없이 수정된 모든 소스코드를 공개하여야 하며, LGPL은 링크시킨 경우 공개의무가 없음. <그림 27>과 같이 링크시킨 경우, LGPL은 파일 A만 공개의무를 가짐



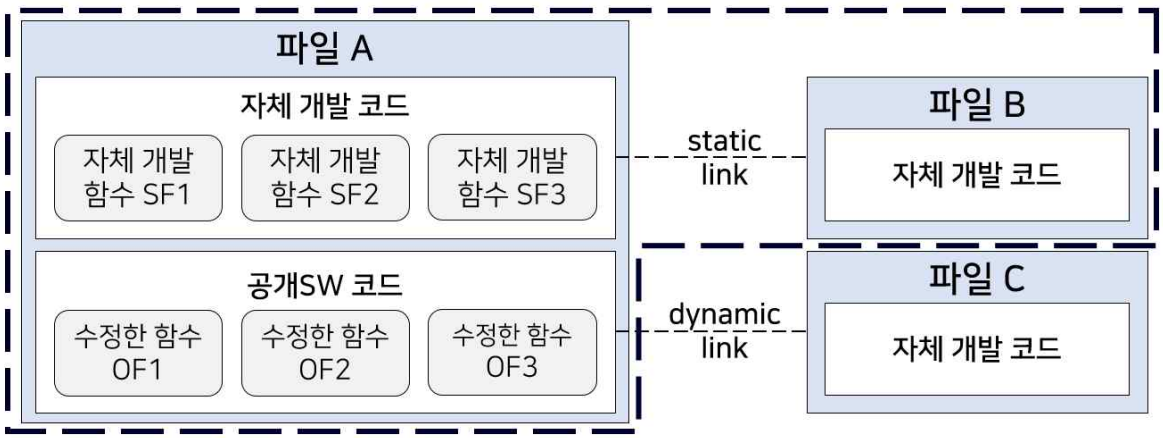
<그림 27> LGPL에서 소스코드 공개 범위 예시

- GPL에서 공개를 원하는 소스코드(재산권 관련 정보 및 기술 등)는 링크와 관계 없이, pipe, socket, command-line argument를 사용하여 통신하는 독립된 프로세스로 구성하는 경우 2차적 저작물 재산권을 확보할 수 있음. <그림 28>과 같이 구성한 경우, GPL은 파일 B, C를 공개하지 않아도 됨



<그림 28> GPL에서 소스코드 공개 범위 예시

- LGPL에서 링크관계를 유지하였다 할지라도, 정적 링크(static link)의 경우는 <그림 29>와 같이 응용프로그램의 목적코드(object)를 공개하여야 함



<그림 29> LGPL에서 응용프로그램 목적코드 공개 범위 예시

- 각 도구에서 사용된 오픈소스 라이선스 간의 차이점을 비교. <표 27>과 같음

구분	GPL v3.0	LGPL v3.0	CDDL v1.0
개요	· GPL 코드를 일부라도 사용하면 전체가 GPL 프로그램이 됨	· GPL을 적용하면 문제가 되는 라이브러리에 적용하기 위해 만든 라이선스 · GPL에 비해 소스 코드의 공개범위가 축소됨	· MPL을 기반으로 썬 마이크로시스템즈에 의해 개조된 오픈소스 라이선스
배포시 소스코드 제공 범위	· fork() 및 exec()를 통해 생성된, 독립된 프로세스로 구성	· 수정코드가 포함되지 않은, 링크된 파일은 공개하지 않아도 됨	· 파일단위 (CDDL이 적용된 코드를 사용한 파일)
2차적 저작물 재산권 확보	· 상동	· 공개를 원치 않는 소스코드는 링크관계를 유지 · 단, 정적링크 시 목적코드는 공개하여야 한다.	· 링크관계를 통하여 2차적 저작물 재산권 확보 가능
비고	· 링크 관계의 소스코드는 정적/동적 관계없이 전체 공개하여야함	· 동적링크 시, 응용프로그램의 목적코드를 공개하지 아니할 수 있음	· CDDL에 의해 원본 및 수정코드를 배포 · 수정한 코드에 대한 소스코드 제공

<표 27> 오픈소스 라이선스 비교

1.3 SIARD 오픈소스 프로젝트 라이선스에 따른 소스코드 공개방법

- 앞서 정책을 비교한 내용을 토대로 각 오픈소스 프로젝트 라이선스에 따른 소스코드 공개방법을 기술함
- 다만, 각 라이선스의 적용을 받는 관련 도구들은 Github를 통한 공개 방법을 취하고 있으므로, 이를 수정하여 재배포하는 경우에도 이러한 방법을 따르는 것이 바람직하다고 보여짐

○ LGPL v3.0 라이선스에 따른 소스코드 공개방법(예, DB preservation toolkit)

- 물리적 매체에 프로그램과 함께 소스코드 동봉 (CD-ROM, USB 등)
- LGPL v3.0 기반의 실행파일과 함께 배포 시, 최소 3년간 소스코드를 제공하겠다는 약정서(Written Offer)를 첨부하여 제공

- 프로그램이 배포되는 동일한 위치에서 소스코드 배포
  ㉠ 인터넷 사이트를 통해서 제품을 배포한다면, 다른 주소의 사이트더라도 동일한 위치에서 배포되는 것으로 간주 (예, Github와 같은 사이트에서 소스코드 제공도 가능)
  ㉡ 라이브러리가 포함되어 있는 프로그램을 FTP 방법으로 제공하고 있는 경우, FTP 서버에서 소스코드를 제공

- P2P 방식으로 제품을 배포한다면, P2P 사용자에게 저작물의 소스코드가 무상으로 공개되는 위치를 공지

○ CDDL v1.0 라이선스에 따른 소스코드 공개방법(예, SIARD Suite의 CDDL v1.0 적용된 소스코드)

- LGPL v3.0 라이선스만큼 소스코드 공개에 대한 세세한 규정이 없음

- 제품을 수취한 자에게 소스코드를 구할 수 있는 방법을 알려주고, 제공자의 사정에 맞는 방식으로 소스코드를 제공

- CDDL v1.0 기반의 수정했다면 그 버전을 직접 제공해야 하고, 수정하지 않은 원본이라면 그 원본을 구할 수 있는 위치를 알려주는 것만으로도 충분

○ 정리: 실제로, 각 라이선스의 적용을 받는 DB preservation toolkit 및 SIARD Suite의 소스코드 또한 각각 KEEP SOLUTIONS 및 스위스연방기록원에 의해, Github를 통한 소스코드 공개정책을 취하고 있어서, 수정·배포하는 경우에도 Github를 통한 소스코드 공개를 취하는 것이 적합하다고 판단됨

1.4 SIARD 해외 활용사례

○ 스위스: SIARD 포맷(eCH-0165), SIARD Suite 및 모범사례 초안(eCH-0233)

▪ 스위스의 eCH 협회에 의해 SIARD 포맷(eCH-0165)이 제정되었으며, SFA(스위스연방기록원)에 의해 SIARD Suite이 개발되어 배포 중 ▪ 현재 스위스 주정부 전자 세금데이터의 보관에 대한 모범사례 초안(eCH-0233)이 작성되었으며, 이 문건에서 SIARD 파일을 생성하기 위한 데이터 모델을 확인할 수 있음

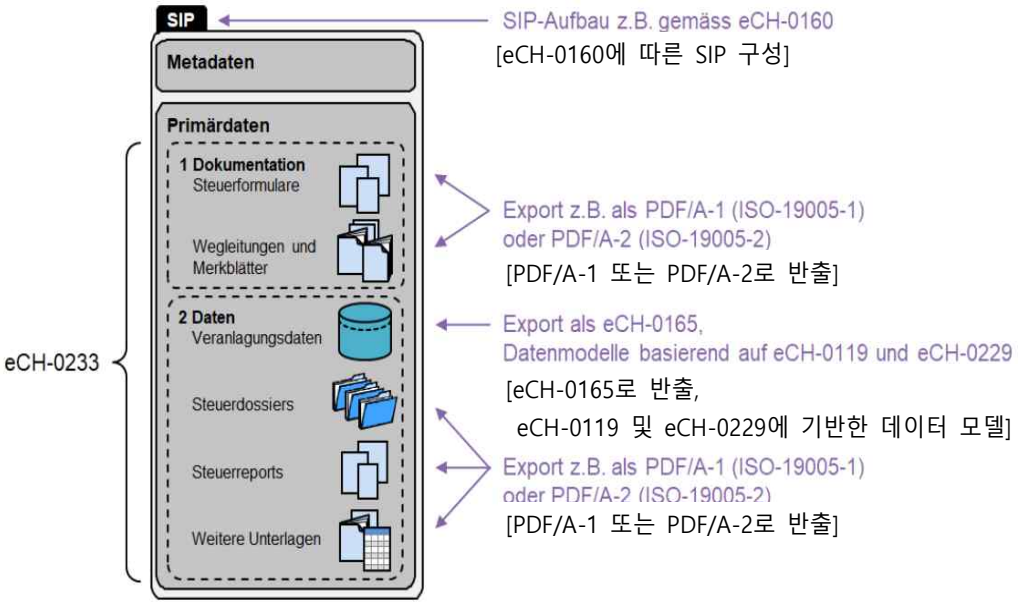
- eCH 협회(eCH Association): 스위스 전자 정부의 활성화를 위한 공공-민간 협력기구
  ※ eCH는 eGovernment Confoederatio Helvetica의 약자로 여기서 Confoederatio Helvetica는 라틴어이며, 실제로 스위스는 이 명칭을 정식명칭으로 사용하고 있으며, 스위스의 국가 도메인도 .ch임

- eCH 협회에서 SIARD 포맷 지침(eCH-0165)을 제정하였으며, SFA(Swiss Federal Archives: 스위스 연방기록원)은 이에 기반하여 SIARD Toolkit인 'SIARD Suite'를 개발, 배포

- eCH-0165: 관계형 데이터베이스의 장기보존을 위한, SIARD 파일 형식의 사양이 기술되

어 있음

- eCH-0233: 주정부 전자 세금데이터의 보관에 대한 모범사례(Best practice)로서 입수정보패키지(SIP: eCH-0160)로 편집된 세금문서를 다룸. 현재 SIARD 파일을 생성하기 위한 데이터 모델 및 SIARD 견본 파일이 첨부되어있는 것으로 파악됨. <그림 30> 참조



<그림 30> eCH-0233 개요

· Metadaten: Metadata(메타데이터), Primärdaten: Primarydata(기본데이터)

· Dokumentation: Documentation(문서), Steuerformulare: Tax forms(세금 양식)

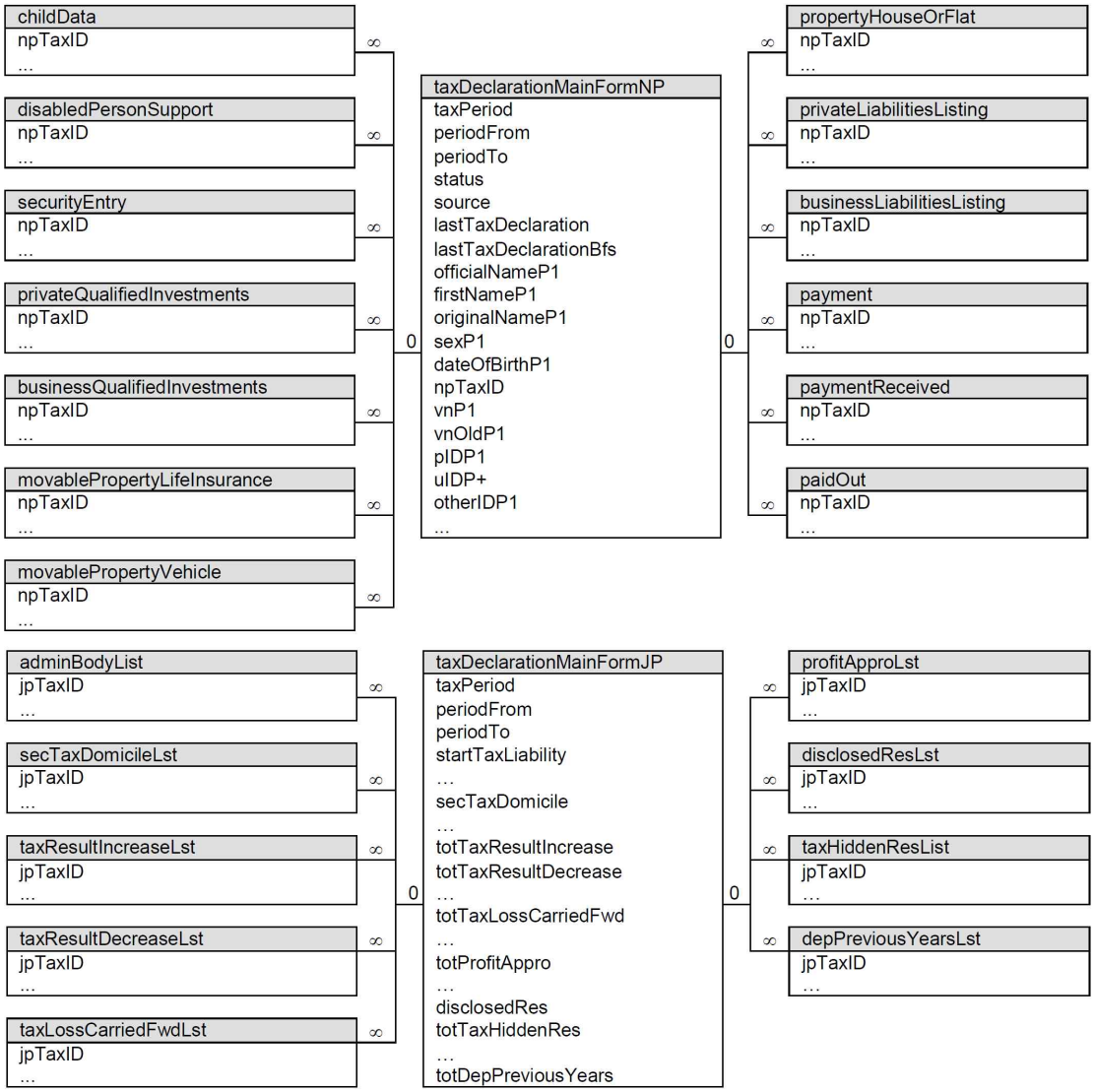
· Wegleitungen und Merkblätter: Guidelines and leaflets(지침 및 전단지)

· Daten: Data(데이터), Veranlagungsdaten: Assessment data(평가 데이터)

· Steuerdossiers: Tax Dossiers(세금 서류), Steuerreports: Tax Reports(세금 보고서)

· Weitere Unterlagen: Further documents(추가 서류)

- 모범사례 eCH-0233에서 전자 세금 신고 지침(eCH-0119, eCH-0229)에 기반한 평가데이터의 디지털 장기 보관을 위해, 위의 그림과 같이 SIARD 포맷 지침(eCH-0165)에 따라 데이터모델 <그림 31>이 작성됨



<그림 31> SIARD 파일 생성을 위해 개인(상)과 법인(하)을 대상으로 한 데이터 모델

· 개인 및 법인의 모든 테이블은 UTF-8로 코딩되어 있음 SIARD파일을 생성하기 위해 eCH-0165의 네임스페이스 및 버전을 채택하여야 함. 구획 확장이 보관된다면, 반드시 각 테이블의 마지막에 있어야 함. 표는 속성 및 요소를 기술하기 위한 열 이름을 보여줌

· eCH-0119: Swiss Tax Conference의 표준 템플릿을 기반으로 개인의 세금 신고 데이터 전송에 대한 표준

· eCH-0229: Swiss Tax Conference의 표준 템플릿을 기반으로 법인의 세금 신고 데이터 전송에 대한 표준

○ 덴마크 국립기록보관소: SIARD-DK(덴마크 SIARD 표준)

- 덴마크 국립기록보관소는 SIARD 1에 기반하여 SIARD-DK를 정보패키지로 사용함
- 스위스의 SIARD 표준과 유사한 구조를 가지고 있으나, 시행령으로서 특정 멀티미디어 포맷의 확장자 및 저장 상세를 별도로 규정함

- 덴마크 국립기록보관소(Danish National Archives)는 SIARD 1에 기반하여 덴마크 시행령(bekendtgørelse) 1007/20(2010)을 사용하기 시작

Lovtidende A

2010 Udgivet den 24. august 2010

20. august 2010. Nr. 1007.

Bekendtgørelse om arkiveringsversioner

<그림 32> 덴마크 시행령(bekendtgørelse) 1007/20(2010)

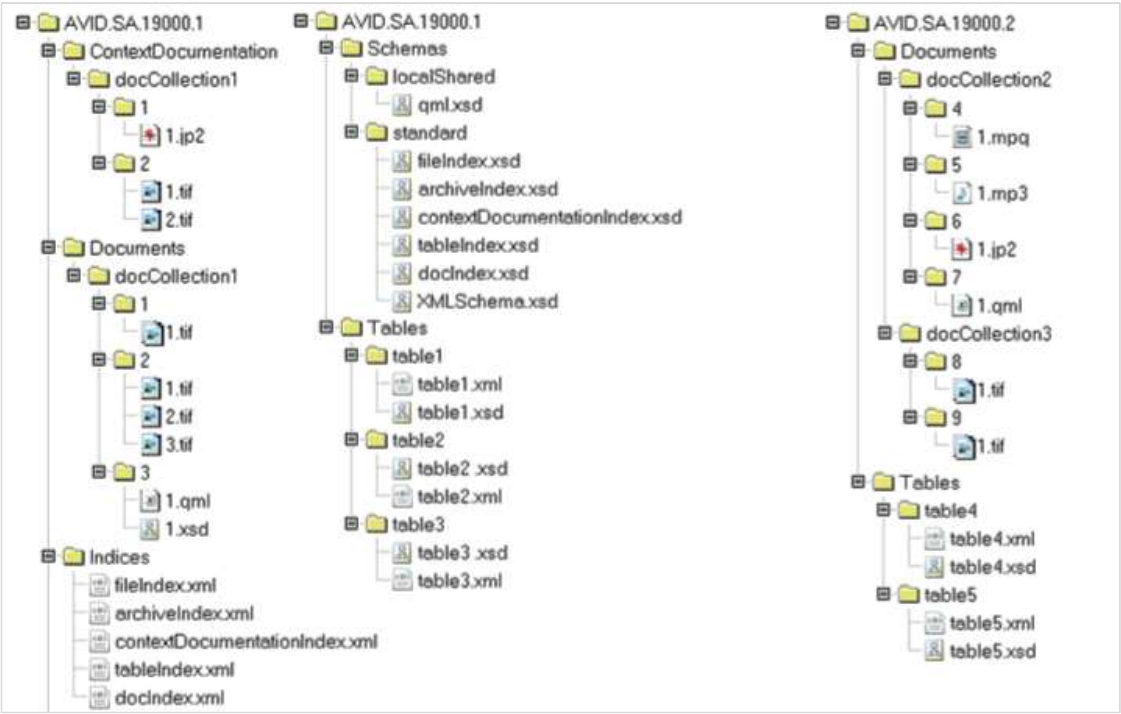
- 덴마크 시행령 1007/20(2010): 정보패키지에 대한 시행령으로서 전자기록입수에 관한 내용이 담겨있음. 이는 국제적으로 SIARD-DK로 알려져 있음
- 덴마크의 SIARD-DK는 SIARD 표준 중 하나이기 때문에, 스위스의 eCH-0165와 유사한 기술구조를 가지고 있음. 다만, 특정 멀티미디어 포맷의 확장자 및 저장 상세를 규정하였음. <표 28>, <표 29> 및 <그림 33>과 같음

멀티미디어 포맷	확장자
TIFF	tif
MP3	mp3
MPEG-2 MPEG-4	mpg
JPEG-2000	jp2
GML(지리적 특성을 표현하기 위한 XML 포맷)	gml
WAVE	wav

<표 28> SIARD-DK 內 멀티미디어 포맷에 대한 확장자 규정

멀티미디어 포맷	저장 상세
TIFF	그래픽 비트맵 TIFF 포맷, version 6.0 baseline (1) 흑백 문서: CCITT / TSS 그룹3, 그룹4, PackBit 또는 LZW로 압축 (2) 그레이 스케일 또는 컬러 문서: PackBit 또는 LZW로 압축
MP3	DS / EN ISO / IEC 11172-3
MPEG-2	DS / EN ISO / IEC 13818-2
MPEG-4	AVC DS / EN ISO / IEC 14496-10 (ITU-T H.264)
JPEG-2000	ISO / IEC 15444-1 : 2004 표준에 따른 JPEG-2000
GML	GML 표준 ISO 19136
WAVE	WAVE LPCM 포맷

<표 29> SIARD-DK 內 멀티미디어 포맷에 대한 저장 상세 요약



<그림 33> SIARD-DK 내의 아카이브 버전 요소 및 구조에 대한 개요

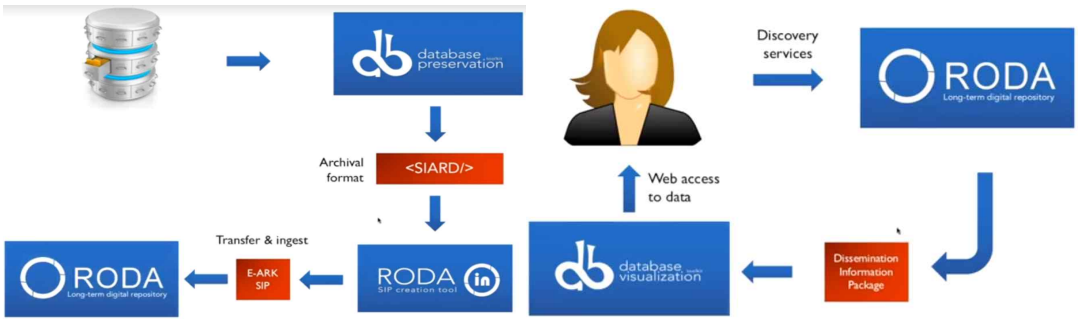
- 2014-2017년에 E-Ark(European Archival Records and Knowledge Preservation) 프로젝트 참여

- E-Ark 프로젝트에서 SFA과 함께 SIARD 2 포맷 개발에 참여하였고, 동시에 SIARD 2 포맷으로 데이터를 추출할 수 있는 DBPTK(DataBase Preservation ToolKit) 개발에도 참여

○ 포르투갈: RODA프로젝트(RODA DBML), KEEP Solutions, DBPTK

- 포르투갈 국립기록원에서는 자체적으로 RODA DBML을 개발하였으나, 현재는 사용하지 않음
- 대신 SIARD 1.0 개발에 참여하였음. KEEP Solutions 사의 오픈소스를 기반으로 SIARD 포맷을 위한 툴킷이 개발되었으며, 이 툴킷의 유지보수는 KEEP Solutions사에서 담당하고 있음

- 포르투갈 국립기록원에서 RODA프로젝트의 일부로 RODA DBML(Database Markup Language)을 개발
- RODA DBML: 데이터베이스를 XML Schema인 DBML으로 마이그레이션한 후, 이를 MySQL에 덤프하고 phpMyAdmin로 시각화하여 접근 권한을 제공함. 현재는 새로운 버전이 개발되지 않고 있음
- 새 버전의 DBML 개발하지 않고, DB의 구조와 컨텐츠를 더 많이 캡처하는 SIARD 1.0 개발에 참여
- RODA repository는 KEEP Solutions 사의 오픈소스 제품
- KEEP Solutions 사는 포르투갈의 Minho 대학(Universidade do Minho)에서 분사된, 디지털 아카이빙 및 디지털 보존 전문회사. SIARD 포맷을 위한 DBPTK 또한 KEEP Solutions가 유지보수하고 있음



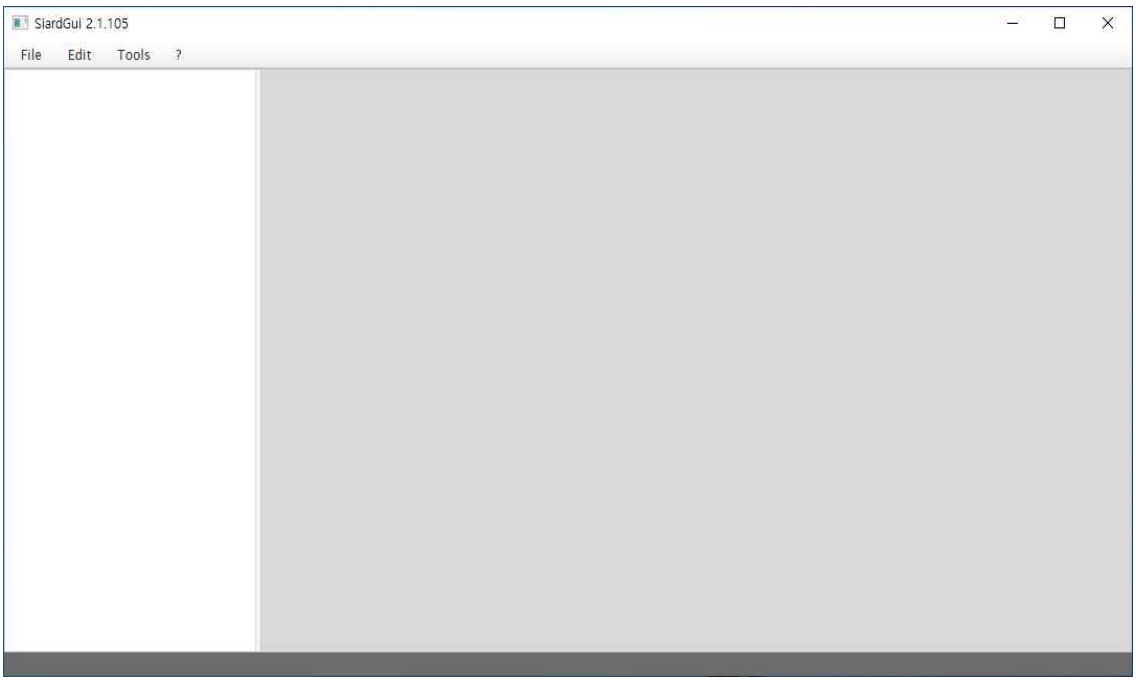
<그림 34> DBPTK와 RODA와의 연계

- DBPTK(Database Preservation Toolkit): 데이터베이스를 디지털로 보존하기 위한 데이터베이스 형식 간 변환이 가능한 툴킷. 기존의 RODA프로젝트에서 독립되어, E-Ark 프로젝트에서 SIARD 2와 함께 추가로 개발됨. 특히 SIARD 2로 보존된 경우, Database Visualization Toolkit을 통해 SIARD 파일의 시각화를 지원
- DBVTK(Database Visualization Toolkit): SOLR를 기반으로 SIARD 2 파일을 탐색, 검색 및 내보내기가 가능한 툴킷. 현재 개발 중

1.5 SiardGui의 메뉴 기능

- 사용자에게 GUI를 제공하는 응용프로그램인 SiardGui의 메뉴 기능을 소개함
- SiardGui는 총 25개의 기능을 제공함

○ SIARD Suite 중에서 사용자에게 GUI(Graphic User Interface)를 제공하는 응용프로그램인 SiardGui의 메뉴는 <그림 35>처럼 “File”, “Edit”, “Tools”, “?” 등 4 메뉴를 제공함



<그림 35> SiardGui 실행화면

○ 4개의 메뉴에 속한 각각의 하위 메뉴는 총 25개이며, 하위 메뉴의 기능은 아래의 <표 30>과 같음

상위 메뉴	하위 메뉴	메뉴 기능
File	Download	· SIARD Suite의 기능을 제공하는 DBMS의 특정 Schema, Tablespace, DB 등을 SIARD 파일로 변환하는 기능
	Recent downloads	· 이용자가 SIARD Suite을 이용해 Download 한 목록 확인
	Upload	· 이용자가 SIARD Suite을 이용해 생성한 SIARD 파일을 DBMS의 DB로 변환하는 기능
	Recent uploads	· 이용자가 SIARD Suite을 이용해 Upload 한 목록 확인

	Open	· SIARD Suite을 이용해 SIARD 파일의 정보(data, column, user)를 확인할 수 있게 보여줌
	Recently Opened	· 이용자가 SIARD Suite을 이용해 Open한 SIARD 파일 목록 확인
	Save	· 이용자가 SIARD Suite에 SIARD 파일을 Open 한 뒤 Database, Schema, Table, Column 등 다양한 계층에서 meta data(description)을 수정했을 경우 수정된 meta data를 포함해 SIARD 파일을 저장하는 기능
	Close	· SIARD Suite에 Open 한 SIARD 파일을 닫는 기능
	Display meta data	· SIARD Suite에 Open 한 SIARD 파일의 meta data 및 Schema, Table 등의 정보를 보여주는 기능
	Import meta data	· 기존 SIARD 파일의 meta data가 저장되어 있을 때, 이용자가 추후 동일한 DB를 SIARD 파일로 변환할 경우 기존 meta data를 이용하는 기능
	Exit	· SIARD Suite을 종료하는 기능
Edit	Copy all	· SIARD Suite에 Open 한 SIARD 파일의 정보를 복사하는 기능
	Copy	· SIARD Suite에 Open 한 SIARD 파일의 정보 중 선택된 정보를 복사하는 기능
	Export table	· SIARD Suite에 Open 한 SIARD 파일의 특정 Table을 html 파일로 추출하는 기능
	Find in meta data	· SIARD Suite에 Open 한 SIARD 파일의 meta data에서 특정 키워드를 검색하는 기능
	Find next in meta data	· SIARD Suite에 Open 한 SIARD 파일의 meta data에서 특정 키워드를 검색했을 때 검색 결과가 복수일 경우, 다음 검색 결과를 볼 수 있는 기능
	Search in table data	· SIARD Suite에 Open 한 SIARD 파일의 특정 Table의 data를 특정 키워드로 검색할 수 있는 기능
	Search next in table data	· SIARD Suite에 Open 한 SIARD 파일의 특정 Table의 data를 특정 키워드로 검색한 결과가 복수일 경우, 다음 검색 결과를 볼 수 있는 기능
Tools	Install	· 설치된 SIARD Suite이 없거나, 설치된 SIARD Suite의 버전이 낮을 경우, 최신 버전의 SIARD Suite 설치를 지원하는 기능
	Uninstall	· SIARD Suite을 제거하는 기능
	Language	· SIARD Suite에서 지원하는 4개 언어(영어, 독일어, 불어, 이태리어)를 선택하는 기능
	Check integrity	· SIARD Suite에 Open한 SIARD 파일의 무결성을 확인하는 기능
	Options	· SIARD Suite의 각종 설정값을 확인, 변경할 수 있는 기능
?	Help	· SIARD Suite의 매뉴얼을 볼 수 있는 기능
	Info	· SIARD Suite 개발과 관련된 정보를 볼 수 있는 기능

<표 30> SiardGui 메뉴 기능 요약표

1.6 공공기관 행정정보시스템 형태·운영·관리 현황

- 행정정보시스템은 데이터베이스 가치, 조직의 기능, 데이터의 처리, 범위, 성격에 따라 유형을 구분할 수 있음. 행정정보데이터세트 또한 첨부문서의 유무 및 데이터 성격에 따라 구분함
- 행정정보시스템은 각 공공기관 별로 독립적으로 데이터 세트를 생산, 관리되고 있음
- 행정정보시스템에서 관리되고 있는 행정정보데이터세트들은 여러 이유(임시방편, 보안, 구현 편의성 등)로 일부 부실하게 되어 기록 관리 측면에서 원하는 정보를 얻을 수 없는 경우가 발생할 수 있음

○ 행정정보시스템의 유형분류

- 국가기록원(2007) 행정정보시스템의 유형을 다음과 같이 구분하고 있음
  - · 데이터베이스 가치에 따라, 인덱스/검색도구, RMS, 통계데이터베이스, 지원시스템으로 구분
  - · 조직의 기능에 따라 조직레벨 시스템, 지식레벨 시스템, 관리레벨 시스템, 전략레벨 시스템으로 구분
  - · 데이터의 처리에 따라 EDMS 유형, OLTP 유형으로 구분
  - · 데이터의 범위에 따라 공동자원 유형, 내부자원 유형으로 구분
  - · 데이터의 성격에 따라 데이터 처리 시스템, 데이터 집계 시스템, 데이터 관리 시스템, 데이터 검색/서비스 시스템으로 구분

○ 행정정보데이터세트의 유형분류

- 또한 행정정보의 유형을 첨부문서의 유무에 따라 구조화된 테이블 형태의 유형과 전자문서를 포함하는 유형으로 구분하고 있으며, 데이터세트의 유형을 데이터 성격에 따라 동적 데이터세트와 정적 데이터세트로 구분하고 있음
- 국가기록원(2015)은 행정정보데이터세트 유형을 데이터 특성, 처리방식, 업데이트 형태에 따라 구분하고 있음
  - · 데이터 특성에 따라 Structured, Semi-Structured, Unstructured 데이터로 구분
  - · 데이터 처리 방식에 따라 OLTP, OLAP, Big Data Analysis로 구분
  - · 데이터 업데이트 형태에 따라 Read-only, Append-only, Continuous-update로 구분
- 조은희, 임진희(2009)는, 아카이빙된 데이터세트의 서비스 특성을 고려하여, 통계 및 설문 등을 수행한 원자료(raw data), 각종 카드 및 대장류, 전자문서와 업무트랜잭션 데이터, 관측데이터 유형으로 구분하고 있음
  - · 통계 및 설문의 원자료 유형은 관계형 데이터베이스에 데이터를 보존하여 분석·활용 서비스 가능함
  - · 카드 및 대장류 유형은 관계형 데이터베이스에 데이터를 보존하되 카드 및 대장 서식의 재현과 증명서 발급 서비스 가능함
  - · 전자문서와 업무트랜잭션 데이터 유형은 사안별로 진행과정을 추적할 수 있도록 전자

문서와 업무트랜잭션 데이터를 기록화함

· 관측데이터 유형은 데이터의 구조와 연관관계를 이해하고 재현할 로직과 어플리케이션을 함께 기록화 하여 활용함

○ 각 개별기관의 독립된 시스템별로 생산, 관리되는 형태로 정보자원 운영의 비효율성 발생

구 분	산림자원통합 관리시스템	국민신문고 시스템	전자연구노트 시스템	특허넷	국토정보 시스템	화학물질종합 정보시스템
운영기관	산림청	국민권익 위원회	한국과학 기술원	특허청	국토교통부	화학물질 안전원
DB 크기	600M	1.2T	2T	15T	3T(원천데이터) 400G(DW/DM)	329G

<표 31> '17.7월 행정정보시스템 내 데이터세트 현황조사 결과

○ 정상명(2017)에 의하면, 2017년 기준 국가기록원이 파악하고 있는 데이터세트가 대량 22,000여 개에 달하지만 지금까지 행정정보시스템에서 생산되고 있는 데이터세트를 전자기록물로 실질적인 관리로. 결국, 시행령에 규정되어 있는 조항은 선언에 그치는 사문화된 조항으로 평가받고 있다고 하였음

○ <표 32>는 DBMS 벤더(상위 5개) 현황

소프트웨어 유형	벤더명	수량(개)	비율(%)
운영체제	Microsoft	14,720	38.82
	RedHat	9,040	23.84
	IBM	6,554	17.28
	HP	3,941	10.39
	CentOS	3,667	9.67
운영체제 합계		37,922	100.00
DBMS	Oracle	12,153	72.35
	Microsoft	3,147	18.74
	티맥스소프트	669	3.98
	큐브리드	484	2.88
	AltiBase	344	2.05
DBMS 합계		16,797	100.00

<표 32> DBMS 벤더(상위 5개) 현황

(출처: 행정안전부, 한국정보화진흥원, 2018)

○ 행정정보시스템에서 관리되고 있는 행정정보데이터세트 관리 문제점

- 전문가 자문, DB개발자 회의, 실데이터 검증을 통해 행정정보데이터세트에 대한 문제점을 확인하여 정리하였음

- 여러 이유(임시방편, 보안, 구현 편의 등)로 행정정보시스템에 있는 데이터세트로부터 기록관리 측면에서 원하는 정보를 얻을 수 없는 경우가 발생하여 추후 활용하기 위해 전자기록을 다시 확인하고자 하여도 해당 기록물의 내용과 맥락 등을 제대로 알 수 없게 됨

- 이러한 상황을 <표 33>에 몇 가지 사례를 정리함. 이러한 문제에 대한 해결을 포함하여 데이터세트 보존포맷 마이그레이션을 수행하기 전에 전처리 과정이 선행이 필요함

<table>
<tr><th>구분</th><th>사례 내용</th></tr>
<tr><td>컬럼(column) 명칭 코드화로 인한 불명확성</td><td>· 컬럼(column) 명칭이 이해하기 어려운 코드로 되어 있으며, 필드(field) 내용도 의미를 알기 힘든 코드로 되어 있거나 잘못된 데이터가 있는 경우도 많음
· 아래 오른쪽 그림처럼 컬럼 명칭이 정해져 있으면 해당 필드 내용을 유추라도 할 수 있지만, 아래 왼쪽 그림처럼 테이블 명칭과 컬럼 명칭이 모두 코드화되어 있으면, 필드 내용의 의미를 알 수 없음
· 보존포맷으로 마이그레이션 하기전에 해당 코드화된 명칭들을 변경하거나, 변경하지 않고 스키마, 테이블, 컬럼 등에 대한 설명을 보존포맷의 메타데이터에 명시하면 해결할 수 있음
Table_00
<table><tr><th>Col_00</th><th>Col_01</th><th>...</th><th>Col_NN</th><th>.....</th></tr><tr><td>20119191</td><td>Jannie Lee</td><td>...</td><td>56473</td><td>...</td></tr><tr><td>20119199</td><td>Donghyun Yang</td><td>...</td><td>78655</td><td>...</td></tr><tr><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td></tr></table>Profiles
<table><tr><th>Student_ID</th><th>Name</th><th>...</th><th>Postal Code</th><th>.....</th></tr><tr><td>20119191</td><td>Jannie Lee</td><td>...</td><td>56473</td><td>...</td></tr><tr><td>20119199</td><td>Donghyun Yang</td><td>...</td><td>78655</td><td>...</td></tr><tr><td>...</td><td>...</td><td>...</td><td>...</td><td>...</td></tr></table></td></tr>
<tr><td>DBMS 상에서 테이블 간 관계를 설정하지 않음</td><td>· DBMS에서는 테이블들을 생성하고, 테이블의 특정 컬럼을 PK(Primary Key)와 FK(Foreign Key)를 설정하고 constraint를 명시하기 때문에 테이블 사이의 관계를 ERD(Entity Relationship Diagram)를 통해 알 수 있는 것이 상식임
· 그러나, 실제로는 구현할 때의 편의성, 추후 시스템 이관의 편리성 때문에 PK, FK, constraint 설정없이 즉, 아무런 관계를 맺지 않은 상태에서 DBMS 내에는 테이블들만 생성하고 데이터만 입력함. 그리고, DBMS에 접속하여 데이터를 조작하는 ‘응용 프로그램’에서 테이블 사이의 관계(PK, FK, constraint 설정 등)를 가정하고 프로그래밍하여 구현하기 때문에 DBMS를 통해 ERD를 추출하여ㅓ는 테이블 사이의 관계를 알 수 없는 경우가 상당히 많음</td></tr>
</table>

▼

· 그렇기 때문에 보존포맷으로 마이그레이션 하기 전에, 기록관리를 위해, 추후 보존된 기록물들을 구체적으로 확인할 수 있도록 데이터세트 관련 정보들이 잘 기록되어야 함
· 스키마, 테이블, 컬럼, 실제 필드안에 들어가는 코드의 의미에 대한 설명은 입력될 필요가 있으며, 또한 테이블 사이의 관계로 별도의 ERD를 통해서 기록되어야 함
· DB유지 및 관리, 응용프로그램 연동을 통해서 DB 접속하여 사용하는 SQL 문장도 같이 기록되어야 함

<표 33> 행정정보시스템의 데이터세트에 존재하는 기록관리 문제점

2. 데이터세트 보존포맷 선정체계 수립 및 보존포맷 선정

- 기록물 생산 환경 및 정보기술의 변화로 다양한 유형의 전자기록이 지속적으로 증가하고 있는 상황에서 일괄적으로 적용하는 현행 단일 문서보존포맷과 장기보존포맷 전략으로는 대응하는 데 한계가 있음
- 다양한 기록유형 및 기술변화 등을 고려하여 전자기록 장기보존의 지속가능성, 유연성, 확장성, 안전성 등을 확보할 수 있는 전자기록 장기보존전략을 개발할 필요가 있음
- 특히, 보존포맷과 관련하여 모든 기록유형에 적용 가능한 보존포맷 선정 기준(공통기준)과 기록유형별 특성을 고려한 보존포맷 선정 기준(고유기준)을 마련함으로써 보존포맷 다양화 전략이 필요
- 이를 위해, 본 연구는 모든 유형에 적용 가능한 보존포맷 선정 기준(공통기준)과 데이터세트 보존포맷 선정 기준(고유기준)을 제안하였으며, 이를 기준으로 보존포맷 적합성 평가체계 개발
  - 공통기준 : 상위기준 총 5개, 하위기준 총 10개 / 데이터세트 고유기준 : 3개
- 본 연구에서 개발한 보존포맷 적합성 평가체계를 적용하여 데이터세트 보존포맷인 'SIARD' 평가검증
  - 데이터세트 보존포맷 적합성 평가 결과 : 'SIARD' - C등급(양호)

2.1 전자기록 보존포맷 선정을 위한 공통기준

- 다양한 기록유형 및 기술변화에 쉽게 대응할 수 있도록 모든 기록 유형에 적용 가능한 전자기록 보존포맷 선정을 위한 공통기준 및 보존포맷 적합성 평가체계 개발
- 전자기록 보존포맷 공통기준은 문헌조사와 파일의 특성을 고려하여 개발함
  - 문헌조사 : 국내·외 아카이브, 도서관, 프로젝트, 연구결과 등에서 제시한 보존포맷 기준
  - 파일의 특성 : 파일 포맷이 만들어지는 과정 및 절차, 파일이 구동되는 원리 및 과정, 파일이 지원하는 기록관리 메타데이터 여부, 파일 사용자 범위, 파일이 제공하는 기록 관련 기능
- 공통기준 제안 : 상위기준 총 5개, 하위기준 총 10개

가. 전자기록 보존포맷 기준 현황

○ 전자기록 보존포맷 선정기준의 필요성

- 기록물 생산 환경과 기술의 변화로 다양한 유형의 전자기록이 생산되고 있지만 현재 국가기록원이 적용하고 있는 보존포맷 전략은 다양한 기록 유형에 적용하기 곤란함

- 전자기록을 영구적으로 보존하고 활용하기 위해서는 현재 포맷이 보존포맷으로 적합한지 검토하고 적합하지 않으면 보존포맷으로 변환해야 하지만 아직은 이에 대한 구체적인 기준이 미흡함

○ 전자기록 보존포맷 기준 현황 분석 대상 및 결과

- 전자기록 보존포맷 기준과 관련하여 국내·외 아카이브, 도서관, 프로젝트, 연구결과 등에서 제시한 내용을 토대로 전자기록 보존포맷 기준에 대한 현황을 분석함

- 보존포맷 기준 관련 용어는 기관 및 연구자에 따라 다르게 사용하고 있어 보존포맷 기준에 대한 정의 및 내용을 기준으로 현황을 분석함(<표 34> 참고)

기준	출처	
	아카이브 및 기타 기관	논문 및 프로젝트
개방성 (Openness)	LAC, TNA, NARA, NAK, LOC, WHS	Abrams et al. 2005; Arms & Fleischhauer, 2006; Barnes, 2006; CENDI, 2007; Clausen, 2004; Eun G.Park & Sam Oh, 2012; Folk & Barkstrom, 2003; Frey, 2000; Hodge & Anderson, 2007; InterPARES 2; Lesk, 1995; Malcolm Todd, 2009; Rog & van Wijk, 2008; Sullivan, 2006; Wijk & Rog, 2007;
안정성 (Stability)	LAC, TNA, NAK, MSA	Brown, 2003; ECMA, 2008; Eun G.Park & Sam Oh, 2012; Folk & Barkstrom, 2003; Frey, 2000; Markus Hamm & Christoph Becker, 2011; Malcolm Todd, 2009; Puglia et al., 2004;
상호운용성 (Interoperability)	LAC, TNA, NAK, LOC, WHS	Abrams et al., 2005; Arms & Fleischhauer, 2006; Brown, 2003; CENDI, 2007; Clausen, 2004; Frey, 2000; Eun G.Park & Sam Oh, 2012; Hodge & Anderson, 2007; Müller et al., 2003; InterPARES 2; Malcolm Todd, 2009; Puglia et al., 2004; Rog & van Wijk, 2008; Sullivan, 2006; Wijk & Rog, 2007;
자체문서화 (Self -Documentation)	TNA, NARA, NAK, LOC, WHS	Abrams et al., 2005; Arms & Fleischhauer, 2006; Barnes, 2006; Brown, 2003; CENDI, 2007; Eun G.Park & Sam Oh, 2012; Folk & Barkstrom, 2003; Hodge & Anderson, 2007; Johnson, 1999; Lesk, 1995; Malcolm Todd, 2009; Müller et al., 2003; Puglia et al., 2004; Rog & van Wijk, 2008; Sullivan, 2006; Wijk & Rog, 2007;
진본성 (Authenticity)	NAK, MSA	Brown, 2003; Eun G.Park & Sam Oh, 2012; Folk & Barkstrom, 2003
채택 (Adoption)	LAC, TNA, NARA, NAK, LOC, WHS	Abrams et al., 2005; Arms & Fleischhauer, 2006; Brown, 2003; CENDI, 2007; Eun G.Park & Sam Oh, 2012; Folk & Barkstrom, 2003; Hodge & Anderson, 2007; InterPARES 2; Malcolm Todd, 2009; Puglia et al., 2004; Rog & van Wijk, 2008; Sullivan, 2006; Wijk & Rog, 2007;
기능성 (Functionality)	NARA, NAK, LOC, MSA, WHS	Abrams et al., 2005; Anderson, 2007; Arms & Fleischhauer, 2006; Brown, 2003; CENDI, 2007; Eun G.Park & Sam Oh, 2012; Folk & Barkstrom, 2003; Frey, 2000; Hodge & Puglia et al., 2004; InterPARES 2; Puglia et al., 2004; Rog & van Wijk, 2008; Sullivan, 2006; Wijk & Rog, 2007;

<표 34> 보존포맷 기준 현황

나. 전자기록 보존포맷 선정을 위한 공통기준 항목

○ 국내 · 외 아카이브, 도서관, 프로젝트, 연구결과 등의 보존포맷 기준들을 종합하여 분석한 결과 보존포맷 기준 총 5개가 도출되었고, 각 기준의 정의와 세부 기준은 <표 35>와 같음

○ 7개의 기준에서 안정성은 보존포맷의 표준화와 관련이 있으며, 표준화는 기본적으로 공유를 목적으로 하기 때문에 개방성의 하위개념으로 볼 수 있음

○ 진본성은 기록에서 반드시 보존되어야 할 특성과 관련되어 있기 때문에 고유기준에서 다뤄야 하기 때문에 공통기준에서는 제외하였음

○ 전자기록 보존포맷 선정을 위한 공통기준 : 상위기준 - 총 5개

no.	기준	정의
1	개방성 (Openness)	• 해당 포맷 및 구동 SW의 소유권에 대한 특정업체의 독점여부 및 해당 포맷의 '표준(standard)'과 '공개코드(Open Source)'의 존재여부를 판단하는 기준 • 세부기준 : 공개가용성(Open Availability), 공표(Disclosure)
2	상호운용성 (Interoperability)	• 해당 포맷과 외부의 다양한 요인(HW, 구동SW, OS 등)과의 독립성, 호환성 및 변환가능성을 판단하는 기준 • 세부기준 : 독립성(Independence), 호환성(Compatibility), 변환가능성(Convertibility)
3	자체문서화 (Self-Documentation)	• 해당 포맷에서 자체 메타데이터 및 기능 지원 여부를 판단하는 기준 • 세부기준 : 메타데이터 지원(Metadata)
4	채택 (Adoption)	• 해당 포맷의 사용 범위를 판단하는 기준 • 세부기준 : 편재성(Ubiquity), 편중성(Sporadicalness)
5	기능성 (Functionality)	• 해당 포맷이 지원하는 다양한 기능(암호화, 오류감지, 검색기능 등)을 판단하는 기준 • 세부기준 : 보호메커니즘(Protection), 검색기능(Retrievability)

<표 35> 전자기록 보존포맷 선정을 위한 공통기준 및 정의

○ 전자기록 보존포맷 선정을 위한 공통기준 : 세부기준 - 총 10개

- **개방성(Openness) 》 <u>공개가용성(Open Availability)</u>**

· 해당 포맷 및 구동 SW가 특정 기업에 '독점'되어 있는지 여부

✓ 해당 포맷을 구동시킬 수 있는 다른 SW 존재여부

✓ 해당 포맷 사용에 대한 제한여부(라이센스, 구독, 특허료 등)

✓ 기본 도구(메모장, 그림판 등) 사용을 통한 분석가능 여부

(예시) 공개가용성 (Open Availability)
1. 해당 포맷 및 구동 SW가 특정 기업에 '독점' 되어 있는 경우 - 기업 : doc/docx, hwp, pdf 등 - 오픈단체 : odf 등 1) 해당 포맷을 구동할 수 있는 다른 SW 존재여부 - 해당 포맷을 구동할 수 있는 SW 개수 2) 해당 포맷 사용에 대한 제한여부(R: Read, W: Write) - 무료 R/ 무료 W : NotePad 등 - 유료 R/ 유료 W : Ultraedit, Micro Office 등 - 무료 R/ 유료 W : Acrobat, 한글과 컴퓨터, Polaris Office 등 3) 기본도구(메모장, 그림판 등) 사용을 통한 분석가능 여부 - 기본 도구를 통해, 해당 포맷을 구성하는 콘텐츠들의 해석 범위 : 전체, 일부, 비율 등 - 텍스트 콘텐츠가 표준 문자 인코딩으로 되어 있는 여부 ※ 표준 문자 인코딩 : UTF-8, 유니코드, 아스키 코드 등 - 압축되어 있는 경우 신뢰성 있는 압축으로 되어 있는지 여부 : docx, pptx, odt 계열 등 ※ 신뢰성 있는 압축 : zip, gzip, lwz 등 - 멀티미디어 콘텐츠가 공개포맷으로 되어 있는지 여부 : jpeg, gif, mpeg 등

- 개방성(Openness) 》 공표(Disclosure)

· 해당 포맷에 대한 '표준(Standard)'의 존재여부

✓ 공개적인 참조 및 이용 여부

✓ 체계적이고 권위 있는 기관에 의한 표준화 여부

· 해당 포맷의 '공개코드(Open Source)' 존재여부

✓ 라이선스 형태(저작권)

(예시) 공표(Disclosure)
1. 해당 포맷의 '표준(Standard)' 존재 여부 1) 공개적인 참조 및 이용 여부 : 다운로드, 열람 등 - 인터넷을 통해 다운로드 할 수 있는지 여부 : 무료, 유료 2) 체계적이고 권위있는 기관에 의한 표준화 여부 - ISO, ITU, OASIS, W3C, IETF, IEEE 등 ※ 국제표준, 국가표준, 단체표준에 따라 차등 점수 부여 2. 해당 포맷의 '공개코드(Open Source)' 존재 여부 : Github, SourceForge, CD/USB 등 1) 라이선스 형태(저작권) : GPL(GNU General Public License), LGPL(Lesser GPL), BSD(Berkeley Software Distribution), MPL(Mozilla Public License), CDDL(Common Development and Distribution License) 등

- 상호운용성(Interoperability) 》 독립성(Independence)

· (OS 관점) 해당 포맷을 구동할 때 특정 OS에 의해 받는 영향도

· (HW 관점) 해당 포맷을 구동할 때 HW에 의해 받는 영향도

· (SW 관점) 해당 포맷 또는 구동 SW가 특정 기술/표준/부가 SW에 의해 받는 영향도

(예시) 독립성(Independence)
1. 해당 포맷을 구동할 때 특정 OS에 의해 받는 영향도 - 특정 OS에서만 실행되는 구동 SW 등 - Unix(HP-UX, Solaris 등), Linux(Centos vO.O, RedhatvO.O, Fedora vO.O등), Windows(3.0, 2000, NT, XP, 7/10 등), iOS 등 2. 해당 포맷을 구동할 때 HW에 의해 받는 영향도 - 특정 HW 필요(예, Bluray 플레이어), 최소 사양, 소형/대형, 제한(x86) 등 3. 해당 포맷 또는 구동 SW가 특정기술/표준/부가 SW에 의해 받는 영향도 - 코덱(codec), 폰트 등

- 상호운용성(Interoperability) 》 호환성(Compatibility)

· 해당 포맷의 현재 구동 SW 지원 여부 및 이전/이후 구동 SW 버전과의 호환 가능 여부(동일한 SW에 한함)

※ 동일한 SW : 같은 제조사, 계열사, 인수회사 등

· 구동하는 SW의 Release 주기(공개주기)에 따른 형식 및 사양의 업데이트 여부

· 해당 포맷의 버전 업데이트 개발 로드맵 또는 계획의 존재 여부

(예시) 호환성(Compatibility)
1. 구동하는 SW의 Release 주기(공개주기)에 따른 형식 및 사양의 업데이트 여부 - 현재 가장 대표성 있는 구동 SW

- 상호운용성(Interoperability) 》 변환가능성(Convertibility)

· 해당 포맷이 정보 손실 없이 다른 포맷으로 변환 가능 여부 및 변환 가능한 포맷의 다양성

· 해당 포맷이 활용하기 위한 목적으로 SW, 서비스 및 도구 등에 의해서 다른 포맷으로 쉽게 변환되고 재사용할 수 있는지 여부

(예시) 변환가능성(Convertibility)
1. 해당 포맷이 정보의 손실없이 다른 포맷으로 변환 가능 여부 및 변환 가능한 포맷의 다양성 - 장기 보존을 위해 추후 안정적인 마이그레이션 보장 가능성 2. 해당 포맷이 활용하기 위한 목적으로 SW, 서비스 및 도구 등에 의해서 다른 포맷으로 쉽게 변환되고 재사용할 수 있는지 여부 - 활용을 위해 해당 포맷을 활용하기 쉬운 포맷으로의 변환 가능 여부(AIP → DIP) ※ FLAC(무손실 압축 음성) → MP3(손실 압축 음성), FFV1(무손실 압축 영상) → MP4(손실 압축 영상) ※ HWP(웹브라우저 뷰어 별도 설치) → PDF(웹브라우저 뷰어 기본 내장)

- 자체문서화(Self-Documentation) 》 메타데이터 지원(Metadata)

· 해당 포맷의 자동 생성 메타데이터 기능 제공 여부

· 해당 포맷의 사용자 정의 메타데이터 기능 지원 여부

· 해당 포맷으로부터 메타데이터 추출 가능 여부

(예시) 메타데이터 지원(Metadata)
1. 해당 포맷의 자동 생성 메타데이터 기능 제공 여부 - 기본적으로 파일 시스템(FAT32, NTFS, ext1/2/3/4 등)에서 자동으로 생성하는 메타데이터 이외에 시스템에서 제공하는 자동 생성 메타데이터 ※ JPG, TIFF의 EXIF(Exchangable Image File), 오피스 계열(pdf, doc(x), ppt(x), hwp(x) 등)의 변경 내용 추적 기능 2. 해당 포맷의 사용자 지정 메타데이터 기능 지원 여부 - 오피스 계열(pdf, doc(x), ppt(x), hwp(x) 등)의 사용자 지정 메타데이터 ※ hwp 파일 오른쪽 마우스 → 속성 → 사용자 지정, pptx 파일 오른쪽 마우스 → 속성 → 자세히 → 속성의 값 수정가능 (Windows 10의 경우) - 오피스 계열(pdf, doc(x), ppt(x), hwp(x) 등)의 메모, java/c의 주석 ※ 메타데이터 기능 : 주석, 메모, 슬라이드 노트 등 3. 해당 포맷으로부터 메타데이터 추출 가능 여부 - java의 javadoc 기능, 오피스 계열(pdf, doc(x), ppt(x), hwp(x) 등)의 메모는 별도 SW 지원, pdf 메타데이터 추출 지원

- 채택(Adoption) 》 <u>편재성(Ubiquity)</u>

· 해당 포맷에 대한 수요와 공급의 법칙이 잘 확립되어 있고 광범위하게 사용되는 포맷인지의 여부

✓ OS에서 별도의 응용 SW의 설치 없이 해당 포맷을 인식하고 내용 확인이 가능한 지 여부

✓ 브라우저에서 별도의 확장 응용 SW의 설치 없이 해당 포맷을 인식하고 내용 확인이 가능한 지 여부

✓ 해당 포맷이 표준화 단체에 의해 표준화 과정을 거쳐 저명한 컨소시엄과 그룹에 의해 채택되어 전 세계에서 사용하고 있는 지 여부

✓ 해당 포맷이 시장을 선도하는 포맷인지 여부

✓ 해당 포맷을 제작/조작/렌더링 할 수 있는 많은 경쟁 제품의 존재 여부

(예시) 편재성(Ubiquity)
1. 해당 포맷에 대한 수요와 공급의 법칙이 잘 확립되어 있고 광범위하게 사용되는 포맷인지의 여부 1) OS에서 별도의 응용 SW의 설치 없이 해당 포맷을 인식하고 내용 확인이 가능한 지 여부 - txt (O) : 메모장/워드패드 등으로 인식 및 내용 확인 가능 - jpeg (O) : 그림판 등으로 인식 및 내용 확인 가능 - xml/html (O) : 메모장 등으로 인식 및 내용 확인 가능 - mp4 (O) : Windows Media Player/영화 및 TV@Windows10 등으로 인식 및 내용 확인 가능 - warc (X) 2) 브라우저에서 별도의 확장 응용 SW의 설치 없이 해당 포맷을 인식하고 내용 확인이 가능한 지 여부 - pdf, xml/html, jpeg (O) : 브라우저에서 인식 가능 - hwp, MS오피스도구, mp4, warc (X) : 브라우저에서 인식 불가능 ※ 브라우저 : Microsoft Edge, Internet Explorer, Chrome, Firefox 등 3) 해당 포맷이 표준화 단체에 의해 표준화 과정을 거쳐 저명한 컨소시엄과 그룹에 의해 채택되어 전 세계에서 사용하고 있는 지 여부 - xml, html (O) : W3C - odt (O) : OASIS - jpg, mp3 (O) : ISO/ITU 4) 해당 포맷이 시장을 선도하는 포맷인지 여부 - hwp(O) : 아래아한글 - MS오피스도구(O), wma(△), wmv(△) : MS 5) 해당 포맷을 제작/조작/렌더링 할 수 있는 많은 경쟁 제품의 존재 여부 - hwp (△) : MS, Polaris 등 - MS오피스도구 (△) : 아래아한글, Polaris 등 - mp3, mp4 (O) : WMP, Gomplayer, 알송 등 - jpg, png, bmp, tiff (O) : 그림판, ACDSEE, 알씨 등

- 채택(Adoption) 》 <u>편중성(Sporadicalnes)</u>

· 해당 포맷이 국립도서관, 기록원 및 기타 기록유산기관이 공식적으로 채택한 보존포맷인지의 여부

· 해당 포맷이 특정 전문 분야의 커뮤니티에서 채택되어 사용되고 있는 포맷인지 여부

- 기능성(Functionality) 》 <u>보호메커니즘(Protection)</u>

· 해당 포맷이 암호 보호, 복사 방지, 디지털 서명, 인쇄 방지 및 컨텐츠 추출 보호와 같은 기술보호메커니즘이 적용되는 포맷인지의 여부

· 해당 포맷의 오류 감지, 수정 메커니즘 및 암호화 옵션의 수용 여부

· 해당 포맷의 우발적인 손상에 대한 탄력성 여부

- 기능성(Functionality) 》 <u>검색기능(Retrievability)</u>

· 문서내용에 대한 검색 기능 제공 여부

다. 전자기록 보존포맷 선정기준 평가체계

○ 전자기록 보존포맷 기준에 대한 내용을 분석한 결과 보존포맷 기준 총 5개, 각 기준에 대한 세부기준 총 10개를 도출하였으며, 이는 전자기록 보존포맷 선정기준의 평가체계를 구축하기 위한 평가요소로서 고려할 수 있음(<표 36>, <표 37> 참고)

○ 전자기록 보존포맷 선정기준 중 **채택**의 **편재성과 편중성은 독립적으로 평가**해야 함

- **편재성**은 **기본적인 평가기준**으로 적용
- **편중성**은 특별한 조건을 충족하는 경우에만 **예외적으로 적용**

공통기준	세부기준	평가항목		Y/N
개방성 (Openness)	1. 공개가용성 (Open Availability)	1.1 특정 기업 외 해당 포맷을 구동시킬 수 있는 다른 SW가 있는가?		Y/N
		1.2 해당 포맷 사용에 대한 제한여부(라이센스, 구독, 특허료 등)	1.2.1 무료 Read인가?	Y/N
			1.2.2 무료 Write인가?	Y/N
		1.3 기본 도구(메모장, 그림판 등) 사용을 통한 분석가능 여부	1.3.1 기본 도구를 통해 해당 포맷을 구성하는 콘텐츠 전체를 해석할 수 있는가?	Y/N
			1.3.2 텍스트 콘텐츠가 표준 문자 인코딩(UTF-8, 유니코드, 아스키 코드 등)으로 되어 있는가?	Y/N
			1.3.3 압축되어 있는 경우 신뢰성 있는 압축(zip, gzip, lwz 등)으로 되어 있는가?	Y/N
			1.3.4 멀티미디어 콘텐츠가 공개 포맷(jpeg, gif, mpeg 등)으로 되어 있는가?	Y/N
	2. 공표 (Disclosure)	2.1 해당 포맷의 '표준' 존재 여부	2.1.1 해당 포맷의 표준을 인터넷 등을 통해 공개적으로 참조 및 이용이 가능한가?	Y/N
			2.1.2 해당 포맷의 표준을 인터넷 등을 통해 공개적으로 참조 및 이용할 때 무료인가?	Y/N
			2.1.3 체계적이고 권위있는 기관에 의해 표준화 과정을 거쳤는가?	Y/N
		2.2 해당 포맷의 '공개 코드' 존재 여부	2.2.1 해당 포맷이 오픈소스 라이선스인가?	Y/N
상호운용성 (Interoperability)	3. 독립성 (Independence)	3.1 OS 관점	3.1.1 해당 포맷을 구동할 수 있는 OS의 개수가 다수인가?	Y/N
		3.2 HW 관점	3.2.1 해당 포맷을 특별한 HW없이 구동할 수 있는가?	Y/N
			3.2.2 해당 포맷을 개인용 컴퓨터 수준의 HW에서 구동할 수 있는가?	Y/N
		3.3 특정 기술, 표준, 부가SW	3.3.1 해당 포맷 또는 구동 SW에 특수 코덱 및 특수 플레이어와 같은 특정 기술이나 부가 SW 등의 영향이 없는가?	Y/N

	4. 호환성 (Compatibility)	4.1 해당 포맷이 현재 구동 SW에서 지원하는가? (동일한 SW(같은 제조사, 계열사, 인수회사 등)에 한함)		Y/N
		4.2 해당 포맷이 이전/이후 구동 SW 버전과 호환이 가능한가? (동일한 SW(같은 제조사, 계열사, 인수회사 등)에 한함)		Y/N
		4.3 해당 포맷은 구동하는 SW의 Release 주기(공개 주기)에 따라 형식이나 사양이 자주 업데이트되는가? (현재 가장 대표성 있는 구동 SW)		Y/N
		4.4 해당 포맷의 버전 업데이트 개발 로드맵 또는 계획이 존재하는가?		Y/N
	5. 변환가능성 (Convertibility)	5.1 보존, 추후 안정적인 마이그레이션 보장 가능성	5.1.1 해당 포맷이 정보의 손실없이 다른 포맷으로 변환 가능한가?	Y/N
			5.1.2 변환 가능한 포맷이 다양한가?	
		5.2 해당 포맷을 활용하기 쉬운 포맷으로 변환가능 여부 (AIP → DIP)	5.2.1 해당 포맷이 SW, 서비스 및 툴과 상호운용되어 새로운 목적으로 콘텐츠를 조작하고 재사용할 수 있는가?	Y/N
자체문서화 (Self - Documentation)	6. 메타데이터 지원 (Metadata)	6.1 해당 포맷이 자동 생성 메타데이터 기능을 제공하는가?		Y/N
		6.2 해당 포맷이 사용자 지정 메타데이터 기능을 제공하는가?		Y/N
		6.3 해당 포맷으로부터 메타데이터를 추출할 수 있는 기능을 지원하는가?		Y/N
채택 (Adoption)	7. 편재성 (Ubiquity)	7.1 OS에서 별도의 응용 SW 설치 없이 해당 포맷을 인식하고 내용을 확인할 수 있는가?		Y/N
		7.2 브라우저(Microsoft Edge, Internet Explorer, Chrome, Firefox 등)에서 별도의 확장 응용 SW 설치 없이 해당 포맷을 인식하고 내용을 확인할 수 있는가?		Y/N
		7.3 해당 포맷이 표준화 단체에 의해 표준화 과정을 거쳐 저명한 컨소시엄과 그룹에 의해 채택되어 전 세계에서 사용하는가?		Y/N
		7.4 해당 포맷이 시장을 선도하는가?		Y/N
		7.5 해당 포맷을 제작/조작/렌더링하는 많은 경쟁 제품의 존재하는가?		Y/N
기능성 (Functionality)	8. 보호메커니즘 (Protection)	8.1 해당 포맷이 암호 보호, 복사 방지, 디지털 서명, 인쇄 방지 및 컨텐츠 추출 보호와 같은 기술보호메커니즘이 적용되어 있지 않은가?		Y/N
		8.2 해당 포맷이 오류 감지, 수정 메커니즘 및 암호화 옵션을 수용하는가?		Y/N
		8.3 해당 포맷이 우발적인 손상에 대한 탄력성이 있는가?		Y/N
	9. 검색기능 (Retrievability)	9.1 해당 포맷이 이용자가 원하는 문서내용에 대한 검색 기능을 제공하는가?		Y/N

<표 36> 전자기록 보존포맷 평가표

공통기준	세부기준	평가항목	Y/N
채택 (Adoption)	1. 편중성 (Sporadicalnes)	1.1 해당 포맷이 국립도서관, 기록원 및 기타 기록유산기관이 공식적으로 채택한 보존포맷인가?	Y/N
		1.2 해당 포맷이 특정 전문 분야의 커뮤니티에서 채택되어 사용되고 있는가?	Y/N

<표 37> 전자기록 보존포맷 평가표 - 예외적 적용

2.2 RDB형 데이터세트 보존포맷 선정을 위한 고유기준

- RDB형 데이터세트 보존포맷 선정을 위한 고유기준 및 보존포맷 적합성 평가체계 개발
- 기록 유형별로 보존되어야 할 정보는 상이하므로 Significant Properties(SP)를 기준으로 RDB형 데이터세트 특성을 도출한 후 이를 기준으로 RDB형 데이터세트 보존포맷 선정을 위한 고유기준 개발
  - SP : 외관(appearance), 내용(content), 맥락(context), 기능(behavior), 구조(structure)
- 고유기준 제안 : 총 3개

가. 데이터세트 특성

○ 국내·외 데이터세트 문헌조사 및 분석으로 도출된 데이터세트 주요 특징은 다음 <표 38>과 같음

출처	데이터세트의 주요 특징
오세라, 박승훈, 임진희. (2018)	“데이터세트에 정형구조의 데이터베이스뿐만 아니라 첨부와 같은 기능에 의해 다양한 포맷의 파일이 포함되어 있다.”
국가기록원 (2007)	“행정정보시스템은 다양한 유형의 DBMS를 사용하고 있다.”
왕호성, 설문원. (2017)	“데이터세트는 수시로 수정되고 활용된다.”
국가기록원 (2015)	“데이터베이스의 규모가 크면 클수록 테이블과 데이터요소는 증가하고 그 관계와 구조를 정의한 스키마 역시 그에 비례하여 더욱 복잡해진다.”
Lindley, Andrew (2013)	“DBMS는 인터페이스를 통해 이 기능을 제공하지만 접근 인터페이스와는 별도로 데이터베이스의 검색과 관리, 데이터베이스 스키마의 생성과 수정 등을 위해 SQL(structured query language) 언어를 사용하여 이 기능들을 조정할 수 있다.”
Ricardo Andre Pereira Freitas (2011)	“데이터세트 내에 비정형 데이터가 존재한다.”
박병주 (2011)	“데이터세트의 사용자 정의 데이터유형(UDT), Large Objects 등을 완벽히 보존해야 한다.”
	“데이터세트는 테이블, 열, 키, 관계 등이 있다.”
	“데이터세트는 View, Function, Trigger, Procedure, 데이터베이스 사용자 권한 등 복잡한 요소가 있다.”

<표 38> 데이터세트의 주요 특징

○ 데이터세트 특성 도출을 위하여 Significant Properties(SP) 도입

- SP의 개념

· InSPECT 프로젝트에서 처음 제시

· 디지털 객체가 접근 가능하고 의미 있는 상태를 유지할 수 있도록 시간 경과에 따라 **보존되어야 하는 디지털 객체의 중요한 특성** (Gareth Knight 2008)

· Essential Characteristics, Significant Characteristics 등 여러 동의어가 존재

- SP의 적용효과

· 전자기록의 SP를 도출하여 이를 보존한다면, **기록의 4대 요건을 유지한 상태**로 보존할 수 있음(TNA 2018)

· NARA, TNA, PLANETS project, NAA 등 **여러 나라에서 연구하고 개발하여 활발히 사용**하고 있음

· 전자기록의 SP를 도출한다면, 향후 **장기보존 전략을 선정할 수 있는 참고자료**로 사용가능

- SP는 다음 <표 39>과 같이 5가지 범주로 구분할 수 있음

Categories	의미
Appearance(Rendering)	· 기록 내의 외형적인 모습을 의미
Behavior	· 기록의 상호작용을 의미
Content	· 기록 내 모든 데이터 및 수식을 의미
Context	· 기록의 메타데이터를 의미
Structure	· 기록의 구조정보 및 외부 정보를 의미

<표 39> Significant Properties의 5가지 범주

- 데이터세트 관점에서 SP를 정의한다면 다음과 같음(Mette van Essen, Maurice de Rooij, Bill Roberts, Maurice van den Dobbelsteen 2011)

· **Appearance(Rendering)** : 접근할 수 있는 응용프로그램에서 데이터세트가 화면에 표시되는 방법

· **Behavior** : 접근할 수 있는 응용프로그램에서 상호작용하는 방법

· **Content** : 주로 데이터베이스 테이블의 내용이지만 데이터가 화면에 표시되는 방법도 포함될 수 있음

· **Context** : 데이터베이스를 사용하는 조직, 비즈니스 프로세스에서 데이터를 사용하는 방법 및 응용 프로그램에서 데이터베이스의 정보를 사용하는 방법

· **Structure** : 데이터베이스의 데이터 - 데이터가 테이블로 구성되고 상호 연결되는 방법

○ 전자문서와 데이터세트의 SP 비교

- 전자문서와 데이터세트의 SP에 대하여 문헌조사를 실시하여 미국의 NARA(National Archives and Records Administration), 영국의 TNA(The National Archives), 플로리다의 FDA(Florida Digital Archive)의 조사 내용을 비교하면 다음 <표 40>와 같음

· 데이터세트의 SP를 조사한 기관은 **미국의 NARA**가 유일함

· **영국의 TNA**는 데이터세트가 아닌 Structed Text, Email, Digital Audio, Raster Images만을 대상으로 조사함

· **Florida Digital Archive Action plan(FDA)와 Archivematica**는 데이터세트를 따로 조사하지 않고, Spreadsheet만 조사함

- **Appearance(Rendering)**에서 전자문서는 레이아웃, 폰트, 컬러 등 외형적인 요소들이 중요한 것에 비해 데이터세트는 중요하게 다루어지지 않고, 스프레드시트만이 전자문서와 같이 외형적인 요소인 셀 형식을 보존하는 것이 중요

- **Behavior**에서 전자문서는 외부와의 연결이 거의 이루어지지 않기 때문에 중요하지 않지만, 데이터세트는 SQL문을 통해 외부와의 질의가 이루어지므로 쿼리 또는 외부 링크가 매우 중요하게 다루어짐

- **Content**에서 전자문서는 문자, 숫자 등 보존하기 어렵지 않은 것으로 이루어져 있기 때문에 중요하게 다루지 않으나, 데이터세트는 데이터 및 수식이 중요함

- **Context**는 메타데이터와 관련된 특성으로 전자문서와 데이터세트의 내용이 비슷함

- **Structure**에서 전자문서와 데이터세트 모두 문자 인코딩, 템플릿, 스키마 등 사전 정의된 구조를 중요하게 보존

SP	전자문서			데이터세트		
	NARA (Text)[3]	TNA (Structed Text)[4]	FDA (Text)[5]	NARA (Database)	TNA (·)	FDA (SpreadSheet)
Appearance (Rendering)	레이아웃/크기, 레이아웃/페이지번호, 레이아웃/비율, 텍스트/폰트, 텍스트/컬러, 컬러	페이지 레이아웃, 텍스트 포맷, 정렬, 높이, 넓이	없음	없음	없음	글꼴, 색상, 크기, 셀 형식 등과 같은 모양 특성을 포함
Behavior	없음	하이퍼링크	없음	쿼리	없음	VBA 매크로 인코딩, 외부 링크
Content	없음	텍스트	모든 문자를 포함한 모든 콘텐츠	없음	없음	모든 셀 데이터 및 수식
Context	설명 메타데이터	제목, 생산자, 생산일자, 메타데이터	없음	설명 메타데이터	없음	작성자, 제목 등
Structure	스키마/연결, 스키마/템플릿, 문자 인코딩, 압축, 비트 심도, 해상도	위치, 속성, Navigation scheme	모든 줄 바꿈 위치	스키마, 문자 인코딩	없음	셀 위치(행, 열) 및 중첩 워크시트와 같은 구조정보

<표 40> 전자문서와 데이터세트의 SP

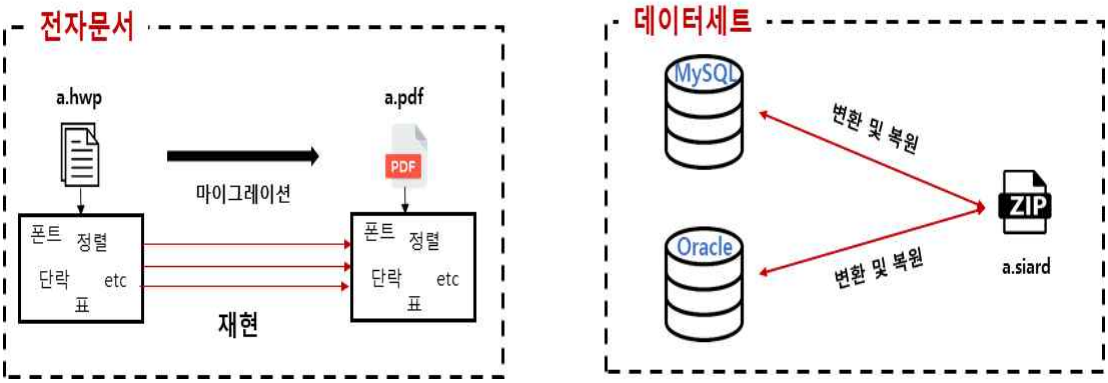
3) https://www.archives.gov/files/era/acera/pdf/significant-properties.pdf

○ 전자문서와 데이터세트의 포맷 변환에 대한 일반적인 특성 비교

- 전자문서와 데이터세트의 특성을 비교하면 <그림 36>과 같이 표현될 수 있음

- 전자문서를 보존포맷으로 변환할 때, Look&Feel이 보존되고, 재현되었는지 즉, 문서가 만들어 졌을 때의 내용뿐만 아니라 모습도 중요함. 그래서 마이그레이션할 때 Look&Feel이 보장되어야 함. 마이그레이션 된 디지털 객체는 새로운 진본이지만, 만일을 대비해서 마이그레이션 되기 전의 원본도 보존함

- 반면, 데이터세트는 Look&Feel 보다 데이터의 콘텐츠와 기능이 더 중요함. 보존포맷으로 마이그레이션 되면 그 디지털 객체는 새로운 진본이 되지만, 원래의 기능을 재현하기 위해서는 데이터세트가 DBMS에 복원되어 활용될 수 있느냐가 가장 중요함



<그림 36> 전자문서와 데이터세트의 포맷 변환에 대한 일반적인 특성

○ SP에서 도출되지 않은 데이터세트 속성(Appearance, Context)

- 데이터세트는 전자문서와 달리 폰트, 레이아웃 등이 중요하지 않고, 데이터 및 기능이 외형보다 훨씬 보존 가치가 높아 Appearance(Rendering) 속성이 도출되지 않음. NARA 역시 대상에서 제외함

- 메타데이터와 관련된 Context 정보를 위해서 국가기록원에서는 데이터세트를 위한 메타데이터인 행정정보시스템 기록관리기준표를 별도로 표준화하고 있기 때문에 본 연구에서는 도출하지 않았음

○ SP와 데이터세트 특징을 비교하여 도출한 데이터세트의 특성은 다음 <표 41>와 같음

4) https://www.kdl.kcl.ac.uk/fileadmin/documents/digifutures/materials/preservation/DF09_prsrv_knight-defiingSigProperties.pdf
5) https://wiki.archivematica.org/Significant_characteristics_of_spreadsheets

SP	특성 설명	데이터세트 특성
Structure	• 관계형 데이터베이스는 기본적으로 Table(Column, Row)로 구성됨 • **테이블 간 관계(Relationship: PK/FK)이 존재**하며, 여러 Table은 하나의 Schema 또는 Database에 포함되기도 함 • 예) Schema, Table, Column, Row, Relation 등	관계성 (Relationship)
	• 대부분 관계형 데이터베이스는 이러한 구조를 가지고 있으며, 이 구조는 반드시 보존되어야 할 필수보존 속성임 • **상용화된 데이터베이스들은 이러한 구조를 각자 다른 설계를 통해 구현**하고 있으며, **데이터베이스는 지속적으로 업데이트되기 때문에 여러 버전들이 존재**함 • 예) Oracle(v5, v6…10g, 11g, 12c..), MySQL(1/2/…/8), SQL Server(2013/2017/2019 등) 등	다양성 (Diversity)
Content	• 데이터베이스의 규모가 클수록 **기능이 다양**해지므로 **관련 데이터세트 요소가 증가하고 복잡**해짐. 이러한 데이터 뿐 아니라 프로시져 등과 같은 루틴(Routine)도 필수보존속성이며 Content 특성에 대응됨 • 예) Privilege, User, Procedure, Table, Function, Role, Trigger, View 등	복잡성 (Complexity)
	• 데이터세트는 정형 데이터뿐만 아니라 전자문서 및 이미지 파일과 같은 **비정형 데이터**, 여러 가지 데이터타입이 데이터세트 내에 포함되므로 필수보존속성이며 Content 특성에 대응됨 • 예) 정수형(INT, SHORT), 실수형(FLOAT, DOUBLE), 문자형(CHAR, VARCHAR), 문장형(STRING, CLOB), 바이너리형(BLOB), 시간형(DATE, TIME) 등	이질성 (Heterogeneity)
Behavior	• 데이터세트는 생산 후, 계속해서 활용되며 **SQL문**을 통하여 데이터를 이용되기 쉽도록 선별 및 조합될 수 있는 있으므로 필수보존 속성의 Behavior 특성에 대응됨 • 예) SELECT, JOIN, CREATE, INSERT 등	상호작용성 (Interactivity)

<표 41> 데이터세트 특성

나. 데이터세트 보존포맷 선정기준 항목

○ 데이터세트의 특성을 통하여 보존포맷 선정기준 수립(<표 42> 참조)

- 일반화(Normalization)

· 상용화된 다양한 종류(제조사, 버전)의 DBMS와의 호환가능성을 판단하는 기준

✓ 보존포맷이 오픈소스일 경우, 지원하지 않는 DBMS를 호환 가능하게 하는 것이 중요

✓ 데이터세트의 특성 중 '다양성'을 고려하여 보존포맷 선정

- 수용성(Acceptability)[6]

· DBMS의 다양한 현재 그리고 미래에 추가될 데이터 타입(정형/비정형) 및 루틴 타입의 수용가능성을 판단하는 기준

✓ DBMS 데이터세트 내 데이터 타입(문자, 숫자, 문장, 이진형 등), 루틴 타입(Stored Procedure, Function, Trigger), External File(비정형 데이터)등을 수용 및 보존해야 함

✓ 데이터세트의 특성 중 '관계성', '복잡성', '이질성'을 고려하여 보존포맷 선정

- 활용성(Usability)[7]

· 변환된 데이터세트 보존포맷의 활용가능성을 판단하는 기준

✓ 보존포맷을 다시 DBMS로 복원하지 않고, 보존포맷 그대로 활용할 수 있어야 함(예, 뷰어)

✓ 데이터세트가 보존포맷에서 재현을 위해 DBMS 복원가능 해야 함

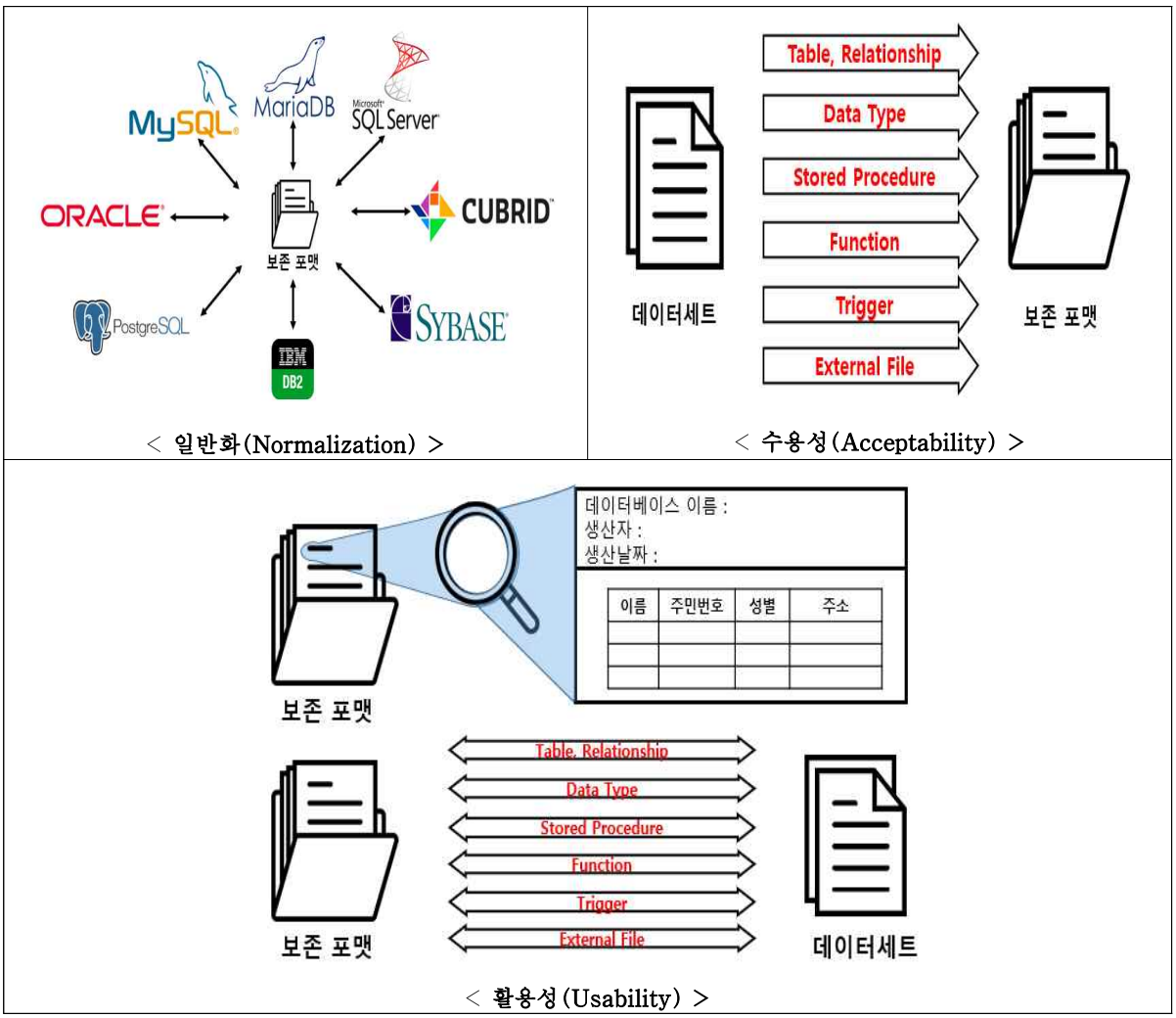
✓ 데이터세트의 특성 중 '상호작용성'을 고려하여 보존포맷 선정

데이터세트 특성	선정기준	내용	
다양성 (Diversity)	일반화 (Normalization)	정의	• 보존포맷은 상용화된 **다양한 종류(제조사, 버전)의 DBMS와 호환**이 가능해야 한다는 기준
		설명	• 보존포맷이 오픈소스일 경우, 지원하지 않은 DBMS를 호환 가능하게 하는 것이 중요
관계성 (Relationship) 복잡성 (Complexity)	수용성 (Acceptability)	정의	• DBMS의 다양한 **현재 그리고 미래에 추가될 데이터 구조 및 관계, 데이터 타입(정형/비정형) 및 루틴 타입**을 수용할 수 있어야 한다는 기준
이질성 (Heterogeneity)		설명	• DBMS 데이터세트 내 테이블 구조 및 관계, 데이터 타입(문자/숫자/문장/이진형 등), 루틴 타입(Stored Procedure, Function, Trigger), External File(비정형 데이터)등을 수용 및 보존해야 함
상호작용성 (Interactivity)	활용성 (Usability)	정의	• 데이터세트를 **보존포맷으로 변환 후 활용 가능**해야 한다는 기준
		설명	• 보존포맷을 다시 DBMS로 복원하지 않고, 보존포맷 그대로 활용할 수도 있어야 함 (예, 뷰어) • 데이터세트가 보존포맷에서 재현을 위해 DBMS 복원가능해야 함

<표 42> 데이터세트 보존포맷 선정을 위한 기준항목 및 설명

6) 수용성(Acceptability)의 경우 아직 용어에 대한 검증이 이루어지지 않았으며, 다른 후보 용어로서 'Convertibility', 'Transferability'등이 있음

7) 활용성(Usability)의 경우, 기록의 4대 속성인 이용가능성(Availability)과 용어가 비슷한 부분이 있으므로 논의 필요



<그림 37> 데이터세트 보존포맷 선정기준 예시

○ 데이터세트 보존포맷 선정기준과 기록의 4대 속성과의 연관성

- Significant Properties는 진본성을 보장하는 특성 도출 틀이기 때문에 선정 기준은 진본성을 보장 가능함

- 수용성과 복원성의 경우, 업무 증거인 첨부파일 보존 또는 복원이 이루어져야 하고, 활용성의 특징인 보존포맷 활용을 통해 업무활동의 신뢰성을 보장할 수 있음

- 무결성은 보존포맷으로 변환되고, 복원될 때, 무결한 상태에서 단계를 진행해야 하기 때문에 수용성, 복원성에서 강하게 관련됨

- 이용가능성은 보존이 제대로 이루어지지 않는다면 이용이 불가능하기 때문에 4가지 항목과 강하게 연관됨

다. 데이터세트 유형 전자기록 보존포맷 평가체계

○ 데이터세트의 특성을 고려하여 보존포맷 선정을 위한 고유기준 총 4개를 도출하였으며, 이는 데이터세트 보존포맷 선정기준의 평가체계 구축을 위한 평가요소로서 고려할 수 있음

○ 데이터세트 보존포맷 선정을 위한 평가 체크리스트 (<표 43> 참고)

- 호환성의 1-3번 항목은 2018/2019년도 정보자원 현황 통계 보고서의 통계에 따라 DBMS 상위 5개를 기준으로 함

고유기준	평가 항목		Y/N
일반화 (Normalization)	1	5개 이상의 DBMS의 데이터세트를 해당 보존포맷으로 변환 가능한가?	Y/N
	2	3개 이상의 DBMS의 데이터세트를 해당 보존포맷으로 변환 가능한가?	Y/N
	3	1개 이상의 DBMS의 데이터세트를 해당 보존포맷으로 변환 가능한가?	Y/N
	4	보존포맷 변환 SW가 오픈소스로 존재하는가?	Y/N
수용성 (Acceptability)	5	보존포맷은 데이터세트의 테이블구조(Column, Row) 및 관계(Relationship)를 보존할 수 있는가?	Y/N
	6	보존포맷은 데이터세트의 데이터타입 계열(문자형, 숫자형, 날짜형, 이진형, 대용량 등)의 데이터를 보존할 수 있는가?	Y/N
	7	보존포맷은 데이터세트의 루틴타입 계열(Stored Procedure, Function, Trigger 등)을 보존할 수 있는가?	Y/N
	8	보존포맷은 데이터세트의 External File을 보존할 수 있는가?	Y/N
활용성 (Usability)	9	보존포맷은 데이터세트의 활용을 위하여 뷰어와 같은 도구를 통해 데이터세트를 확인할 수 있는가?	Y/N
	10	보존포맷은 데이터세트의 활용을 위하여 뷰어와 같은 도구를 통해 SQL 수행이 가능한가?	Y/N
	11	보존포맷은 원래의 DBMS[8]로 테이블구조(Column, Row) 및 관계(Relationship)를 복원할 수 있는가?	Y/N
	12	보존포맷은 원래 생성된 DBMS로 데이터타입 계열(문자형, 숫자형, 날짜형, 이진형, 대용량)을 복원할 수 있는가?	Y/N
	13	보존포맷은 원래 생성된 DBMS로 루틴타입 계열(Stored Procedure, Function, Trigger 등)을 복원할 수 있는가?	Y/N
	14	보존포맷은 원래 생성된 DBMS로 External File을 복원할 수 있는가?	Y/N
	15	보존포맷은 원래 생성된 DBMS가 아닌 다른 DBMS로 복원할 수 있는가?	Y/N

<표 43> 데이터세트 보존포맷 평가표

라. 전자기록 보존포맷 선정을 위한 평가체계

○ 평가목적

- 다양한 기록유형 및 기술변화 등을 고려하여 전자기록 장기보존의 지속가능성, 유연성, 확장성, 안전성 등을 확보할 수 있는 전자기록 장기보존전략을 위한 기초 자료 제공

- 모든 기록유형에 적용 가능한 보존포맷 선정 기준을 제안하고, 이를 기준으로 전자기록 보존포맷 평가체계를 개발함으로써 전자기록 보존포맷 다양화 전략을 위한 방안 제시

○ 평가근거

- 법적근거 : 전자기록 보존포맷 선정을 위한 평가는 공공기록물 관리에 관한 법률 제20조, 동법 시행령 제36조, 제46조를 근거로 실시

- 정책근거 : 국가기록원 전자기록물 장기보존 정책(안)을 근거로 실시

○ 평가대상

- 전자기록 보존포맷 선정을 위한 파일 유형별 보존포맷 평가대상 예시는 <표 44>과 같음

유형		문서보존포맷
문서(텍스트)		EPUB, PDF(PDF/A, PDF/A-1, PDF/A-2), DOCX, TXT(ASCⅡ, Unicode), ODF, OOXML. RTF, DOC, ODT, 기타
프리젠테이션		ODP, PDF(PDF/A-1, PDF/A-2), PPT, PPTX, 기타
데이터 세트	Non-DB Type	JSON, CSV, XLS, XLSX, ODS, TXT(ASCⅡ, Unicode), XML, EBCDIC, 기타
	DB Type	DBF, SIARD, 기타
정적이미지		TIFF, JP2(JPEG2000), DNG(Digital Negative), BMP, GIF, JPG, PNG, PSD, EPS(Encapsulated Postscript), PDF(PDF/A, PDF/A-1, PDF/A-2), ODG(OTG), SVG., 기타
오디오		WAV/WAVE, BWF, FLAC, AIFF, MP3, 기타
동영상		DPX, AVI, WMV, MPG(MPEG-2, MPEG-4), MXF, DCDM, MOV, MP4, 기타
웹		WARC, ARC, HTML, 기타
이메일		EML, MBOX, PST, MSG, 기타
CAD		STEP, PDF/E, DXF, 기타
지리공간		GML, GTIFF, ESRI ArcInfo Export(E00), TerraGo Geospatial PDF, ESRI SHP(ESRI Shapefile), 기타

<표44> 전자파일 유형별 보존포맷 평가대상 예시

8) 원래의 DBMS는 최초로 생성되었던 데이터세트가 관리되었던 DBMS와 동일한 기종이며, 해당 기종의 버전을 지원하는 DBMS를 의미함

- 보존포맷으로서 평가대상 우선순위 선정

· 보존포맷을 선호포맷과 허용포맷으로 구분하여 제안한 국립아카이브 3개 기관[9]을 중심으로 분석

· 파일유형별 보존포맷 선정을 위한 평가대상 우선순위를 '상', '중', '하'로 구분하였으며, 구체적인 예는 <표 45>와 같음

✓ 상 : 선호포맷 중 2개 기관 이상이 선택한 경우 - 보존포맷으로서 최우선 평가대상

✓ 중 : 선호포맷 중 1개 기관이 선택 / 허용포맷 중 2개 기관 이상이 선택한 경우

✓ 하 : 허용포맷 중 1개 기관 선택한 경우

구분	상	중	하
문서(텍스트)	PDF/A-1, PDF/A-2, TXT	EPUB, ODF, ODT, PDF, DOCX, DOC	EPUB, OOXML
프리젠테이션	PDF/A-1, ODP	PDF/A-2, PPT, PPTX	PDF/A-2
데이터세트	ASCII, CSV	JSON, XML, ODS, XLS, XLSX, EBCDIC	JSON, CSV, ASCII, XML, DBF, Unicode, SIARD, MS Access
정적이미지	TIFF, JP2, PNG	ODG(OTG), PDF/A, PDF/A-1, JPG, PDF/A-2, SVG, DNG, GIF, JPG	ODG(OTG), DICOM, Exif
오디오	BWF	FLAC, WAV/WAVE, AIFF, MP3	FLAC, AAC, MPEG-1 layer3, MPEG-2 layer3(MP3enc, Lame Codec), MPEG4(AAC codec)
동영상	DPX	DCDM, AVI, MXF, MOV, DCP, WMA, MPG	DCDM
웹	WARC	ARC, XHTML	WARC, HTML
이메일	EML, MBOX	PST, MSG	EML, MBOX, XML
CAD		STEP, X3D, DXF, AutoDesk's Drawing File, PDF/E	U3D, PRC, DWG
지리공간	GML, GTIFF, KML	ESRI SHP, TIGER, BIL, BIP, BSQ, DEM, ESRI Arc/Info ASCII Grid, ESRI ArcInfo Export (E00), TerraGo Geospatial PDF	GML, GTIFF, ESRI SHP, Vector Product Format, SDTS, CCOGIF, DIG3, Geospatial PDF, IHO S-57

<표 45> 전자파일 유형별 보존포맷 평가대상 우선순위 예시

9) 미국 NARA, 캐나다 LAC, 호주 NAA

○ 전자기록 보존포맷 선정을 위한 절대평가 절차

- (1단계) 공통기준 적합성 평가 : 전자기록 보존포맷 선정을 위한 '공통기준' 평가표를 적용하여 전자기록 보존포맷으로서 적합성 평가

- (2단계) 고유기준 적합성 평가 : 특정 유형의 전자기록 보존포맷 선정을 위한 '고유기준' 평가표를 적용하여 특정유형 전자기록 보존포맷으로서 적합성 평가

- (3단계) 1·2단계 평가 점수를 합산한 후 평점으로 환산(100%)하여 등급을 부여한 다음 최종 보존포맷으로서 적합성 평가

- (4단계) 부적합 평가를 받은 보존포맷의 경우 예외적 규정(편중성)의 적용여부 판단하여 최종 결정

○ 전자기록 보존포맷 선정을 위한 평가방법

- 평가등급 : 5개 등급(A~E)

- 평가점수 : 100점 만점으로 환산하여 평점이 90% 이상인 경우는 'A등급(매우 우수)', 80% 이상인 경우는 'B등급(우수)', 70% 이상인 경우는 'C등급(양호)', 60% 이상인 경우는 'D등급(보통)', 60% 미만인 경우는 'E등급(미흡)'으로 정함(<표 46> 참조).

- 절대평가 : 전자기록 보존포맷 선정을 위한 보존포맷 후보군 도출(<표 46> 기준)

- 상대평가 : 보존포맷 후보군(절대평가 결과) 중 등급에 따라 전자기록 보존포맷 선정을 위한 우선순위 결정

등급	평점(환산점수)	수준정의
A (매우 우수)	90 이상	• 매우 높은 수준의 안정적인 전자기록 보존포맷 • 보존포맷 적합성 : 적합 - 10년마다 재평가 실시하여 등급 재설정
B (우수)	80 이상 (80이상 ~ 90미만)	• 높은 수준의 전자기록 보존포맷이지만 정기적인 평가 필요 • 보존포맷 적합성 : 적합 - 5년마다 재평가 실시하여 등급 재설정
C (양호)	70 이상 (70이상 ~ 80미만)	• 전자기록 보존포맷으로 선정하기에는 다소 미흡한 부분이 있으므로 보존포맷 선정 여부는 상대평가로 결정 • 보존포맷 적합성 : 부분적합 - B 등급 이상의 보존포맷이 없거나 적은 경우 채택 - 3년마다 재평가 실시하여 등급 재설정
D (보통)	60 이상 (60이상 ~ 70미만)	• 전자기록 보존포맷으로 선정하기에는 상당히 미흡한 부분이 있으므로 보존포맷 선정 여부는 상대평가로 결정 • 보존포맷 적합성 : 부분적합 - C 등급 이상의 다른 보존포맷이 없는 경우에만 채택 - 3년마다 재평가 실시하여 등급 재설정
E (미흡)	60 미만	• 전자기록 보존포맷으로서 매우 미흡하므로 선정 불가 • 보존포맷 적합성 : 부적합

<표 46> 전자기록 보존포맷 등급 및 평점 기준

마. 데이터세트 유형 전자기록 보존포맷 제시(권고포맷)

○ 본 연구에서 개발한 전자기록 보존포맷 선정을 위한 평가체계를 적용하여 RDB형 데이터세트의 보존포맷 중 'SIARD'를 대상으로 보존포맷으로서의 적합성 검증(<표 47>, <표 48>, <표 49> 참조)

공통기준		평가항목		Y/N	점수
개방성	1. 공개가용성	1.1 특정 기업 외 해당 포맷을 구동시킬 수 있는 다른 SW가 있는가?		Y	1
		1.2 해당 포맷 사용에 대한 제한여부(라이센스, 구독, 특허료 등)	1.2.1 무료 Read인가?	Y	1
			1.2.2 무료 Write인가?	Y	1
		1.3 기본 도구(메모장, 그림판 등) 사용을 통한 분석가능 여부	1.3.1 기본 도구를 통해 해당 포맷을 구성하는 콘텐츠 전체를 해석할 수 있는가?	Y	1
			1.3.2 텍스트 콘텐츠가 표준 문자 인코딩(UTF-8, 유니코드, 아스키 코드 등)으로 되어 있는가?	Y	1
			1.3.3 압축되어 있는 경우 신뢰성 있는 압축(zip, gzip, lwz 등)으로 되어 있는가?	Y	1
			1.3.4 멀티미디어 콘텐츠가 공개 포맷(jpeg, gif, mpeg 등)으로 되어 있는가?	N	0
	2. 공표	2.1 해당 포맷의 '표준' 존재 여부	2.1.1 해당 포맷의 표준을 인터넷 등을 통해 공개적으로 참조 및 이용이 가능한가?	Y	1
			2.1.2 해당 포맷의 표준을 인터넷 등을 통해 공개적으로 참조 및 이용할 때 무료인가?	Y	1
			2.1.3 체계적이고 권위있는 기관에 의해 표준화 과정을 거쳤는가?	Y	1
		2.2 해당 포맷의 '공개코드' 존재 여부	2.2.1 해당 포맷이 오픈소스 라이선스인가?	Y	1
상호 운용성	3. 독립성	3.1 OS 관점	3.1.1 해당 포맷을 구동할 수 있는 OS의 개수가 다수인가?	Y	1
		3.2 HW 관점	3.2.1 해당 포맷을 특별한 HW없이 구동할 수 있는가?	Y	1
			3.2.2 해당 포맷을 개인용 컴퓨터 수준의 HW에서 구동할 수 있는가?	Y	1
		3.3 특정 기술, 표준, 부가SW	3.3.1 해당 포맷 또는 구동 SW에 특수 코덱 및 특수 플레이어와 같은 특정 기술이나 부가 SW 등의 영향이 없는가?	N	0
	4. 호환성	4.1 해당 포맷이 현재 구동 SW에서 지원하는가?		Y	1

		(동일한 SW(같은 제조사, 계열사, 인수회사 등)에 한함)			
		4.2 해당 포맷이 이전/이후 구동 SW 버전과 호환이 가능한가? (동일한 SW(같은 제조사, 계열사, 인수회사 등)에 한함)		Y	1
		4.3 해당 포맷은 구동하는 SW의 Release 주기(공개 주기)에 따라 형식이나 사양이 자주 업데이트되는가? (현재 가장 대표성 있는 구동 SW)		Y	1
		4.4 해당 포맷의 버전 업데이트 개발 로드맵 또는 계획이 존재하는가?		N	0
	5. 변환가능성	5.1 보존, 추후 안정적인 마이그레이션 보장 가능성	5.1.1 해당 포맷이 정보의 손실없이 다른 포맷으로 변환 가능한가?	Y	1
			5.1.2 변환 가능한 포맷이 다양한가?		
		5.2 해당 포맷을 활용하기 쉬운 포맷으로 변환 가능 여부 (AIP → DIP)	5.2.1 해당 포맷이 SW, 서비스 및 툴과 상호 운용되어 새로운 목적으로 콘텐츠를 조작하고 재사용할 수 있는가?	N	0
자체 문서화	6. 메타데이터 지원	6.1 해당 포맷이 자동 생성 메타데이터 기능을 제공하는가?		Y	1
		6.2 해당 포맷이 사용자 지정 메타데이터 기능을 제공하는가?		Y	1
		6.3 해당 포맷으로부터 메타데이터를 추출할 수 있는 기능을 지원하는가?		Y	1
채택	7. 편재성	7.1 OS에서 별도의 응용 SW 설치 없이 해당 포맷을 인식하고 내용을 확인할 수 있는가?		N	0
		7.2 브라우저 (Microsoft Edge, Internet Explorer, Chrome, Firefox 등)에서 별도의 확장 응용 SW 설치 없이 해당 포맷을 인식하고 내용을 확인할 수 있는가?		N	0
		7.3 해당 포맷이 표준화 단체에 의해 표준화 과정을 거쳐 저명한 컨소시엄과 그룹에 의해 채택되어 전 세계에서 사용하는가?		N	0
		7.4 해당 포맷이 시장을 선도하는가?		Y	1
		7.5 해당 포맷을 제작/조작/렌더링하는 많은 경쟁 제품의 존재하는가?		N	0
기능성	8. 보호메커니즘	8.1 해당 포맷이 암호 보호, 복사 방지, 디지털 서명, 인쇄 방지 및 컨텐츠 추출 보호와 같은 기술보호메커니즘이 적용되어 있지 않은가?		N	0
		8.2 해당 포맷이 오류 감지, 수정 메커니즘 및 암호화 옵션을 수용하는가?		N	0
		8.3 해당 포맷이 우발적인 손상에 대한 탄력성이 있는가?		N	0
	9. 검색기능	9.1 해당 포맷이 이용자가 원하는 문서내용에 대한 검색 기능을 제공하는가?		Y	1
합계					22/33

<표 47> 전자기록 보존포맷으로서 공통기준 적합성 평가 : SIARD

고유기준	평가 항목		Y/N	점수
일반화	1	5개 이상의 DBMS의 데이터세트를 해당 보존포맷으로 변환 가능한가?	Y	1
	2	3개 이상의 DBMS의 데이터세트를 해당 보존포맷으로 변환 가능한가?	Y	1
	3	1개 이상의 DBMS의 데이터세트를 해당 보존포맷으로 변환 가능한가?	Y	1
	4	보존포맷 변환 SW가 오픈소스로 존재하는가?	Y	1
수용성	5	보존포맷은 데이터세트의 테이블구조(Column, Row) 및 관계(Relationship)를 보존할 수 있는가?	Y	0.5[10]
	6	보존포맷은 데이터세트의 데이터타입 계열(문자형, 숫자형, 날짜형, 이진형, 대용량 등)의 데이터를 보존할 수 있는가?	Y	1
	7	보존포맷은 데이터세트의 루틴타입 계열(Stored Procedure, Function, Trigger 등)을 보존할 수 있는가?	N	0
	8	보존포맷은 데이터세트의 External File을 보존할 수 있는가?	Y	1
활용성	9	보존포맷은 데이터세트의 활용을 위하여 뷰어와 같은 도구를 통해 데이터세트를 확인할 수 있는가?	Y	1
	10	보존포맷은 데이터세트의 활용을 위하여 뷰어와 같은 도구를 통해 SQL 수행이 가능한가?	N	0
	11	보존포맷은 원래의 DBMS로 테이블구조(Column, Row) 및 관계(Relationship)를 복원할 수 있는가?	Y	1
	12	보존포맷은 원래 생성된 DBMS로 데이터타입 계열(문자형, 숫자형, 날짜형, 이진형, 대용량)을 복원할 수 있는가?	Y	1
	13	보존포맷은 원래 생성된 DBMS로 루틴타입 계열(Stored Procedure, Function, Trigger 등)을 복원할 수 있는가?	N	0
	14	보존포맷은 원래 생성된 DBMS로 External File을 복원할 수 있는가?	Y	1
	15	보존포맷은 원래 생성된 DBMS가 아닌 다른 DBMS로 복원할 수 있는가?	Y	1
합계				11.5

<표 48> RDB형 데이터세트 보존포맷으로서 고유기준 적합성 평가 : SIARD

구분	평가내용		합계
	공통기준	고유기준	
점수(총점)	22(33)	11.5(15)	33.5(48)
평점(100%)	67(100)	77(100)	70(100)
등급	D(보통)	C(양호)	C(양호)
최종 평가 결과	부분적합		

<표 49> RDB형 데이터세트 보존포맷으로서 최종 적합성 평가결과 : SIARD

10) Orcle에서는 관계 보존이 안되므로 부분점수 부여

2.3 Non-DB형 데이터세트 보존포맷 선정을 위한 고유기준

- Non-DB형 데이터세트의 보존포맷 선정을 위해 대표적인 포맷인 BIFF, CSV, XML, XML DTD 1.0을 대상으로 하였으며, Significant Properties(SP)에 의거하여 각 포맷별 특징을 알아보고 데이터세트 관점에서 특성을 도출함
- 도출된 특성에 따라 Non-DB형 데이터세트의 보존포맷 별 평가표를 작성하였으나, 이에 대해서는 추가적인 검토가 필요할 것으로 보여짐

가. 기준항목

○ Non-DB형 데이터세트는 전자문서가 첨부되지 않고, 테이블 형식의 구조화 된 데이터의 특징을 가진 집합이며, 크게 실질적인 내용을 담고 있는 테이블들과 메타데이터로 구성

- 기록에 대한 내용정보를 갖고 있는 테이블들은 각각의 로우(row)와 컬럼(column)값으로 이루어짐

- Non-DB형 데이터세트는 Oracle, MySQL, 큐브리드 등의 상용 DB 포맷으로 구성된 데이터세트를 제외한 데이터세트를 지칭함. BIFF와 CSV, XML, XML DTD 1.0 동의 포맷들이 여기에 해당함

○ 데이터세트 관점에서 Significant Properties를 정의한다면 아래 <표 50>와 같음(Mette van Essen, Maurice de Rooij, Bill Roberts, Maurice van den Dobbelsteen 2011)

Categories	의미
Appearance(Rendering)	· 기록 내의 외형적인 모습을 의미
Behavior	· 기록의 상호작용을 의미
Content	· 기록 내 모든 데이터 및 수식을 의미
Context	· 기록의 메타데이터를 의미
Structure	· 기록의 구조정보 및 외부 정보를 의미

<표 50> Significant Properties의 5가지 범주

· Appearance(Rendering) : 접근할 수 있는 응용프로그램에서 데이터세트가 화면에 표시되는 방법

· Behavior : 접근할 수 있는 응용프로그램에서 상호작용하는 방법

· Content : 주로 데이터세트의 내용이지만 데이터가 화면에 표시되는 방법도 포함될 수 있음

· Context : 데이터세트를 사용하는 조직, 비즈니스 프로세스에서 데이터를 사용하는 방법 및 응용 프로그램에서 데이터베이스의 정보를 사용하는 방법

· Structure : 데이터세트의 데이터-데이터가 테이블로 구성되고 상호 연결되는 방법

○ Non-DB형 전자기록의 Significant Properties(SP)

- Non-DB형 전자기록의 Significant Properties에 대하여 문헌조사를 실시하여 FDA(Florida Digital Archive)의 조사 내용을 정리하면 다음 <표 51>과 같음

SP	Non-DB형 전자문서 포맷			
	BIFF8	CSV	XML 1.0	XML DTD 1.0
Appearance	글꼴, 색상, 크기, 셀 형식 등과 같은 모양 특성을 포함	없음	없음	없음
Behavior	VBA 매크로 인코딩, 외부 링크	없음	XML 문서에서 참조하는 외부 스키마	DTD 파일에 정의된 제약조건
Content	모든 셀 데이터 및 수식	셀 데이터의 내용, 선택적 CSV 헤더	모든 텍스트 내용, <CDATA> 내에 정의 된 데이터	없음
Context	작성자, 제목 등	없음	XML 문서에 기술된 맥락 정보	없음
Structure	셀 위치(행, 열) 및 중첩 워크시트와 같은 구조 정보	행과 열을 포함한 CSV의 구조 요소	XML 문서에서 정의된 트리와 같은 요소 구조	DTD 파일에 정의된 문서 구조

<표 51> Non-DB형 전자문서 포맷의 SP(FDA)

- Appearance에서 BIFF8만이 전자기록으로서 외형적인 요소인 셀 형식을 보존하는 것이 다루어지고 있음.

- Behavior에서 Non-DB형 전자문서는 DB형 데이터세트와 같이 SQL문을 통한 질의가 이루어지지 않지만, 부수적인 VBA 매크로 인코딩 및 외부 링크, 참조된 외부 스키마, 정의된 제약조건 등을 보존 대상으로 함

- Content에서 Non-DB형 전자문서는 데이터 및 수식을 중요하게 다룸.

- 메타데이터와 관련된 특성인 Context는 BIFF8 및 XML 1.0의 내용이 비슷함. CSV 및 XML DTD 1.0은 해당 사항을 다루지 않음

- Structure에서 Non-DB형 전자문서는 모두 셀 위치(행, 열), 문자 인코딩, 템플릿, 스키마 등 사전 정의된 구조를 중요하게 보존

○ Significant Properties에서 도출되지 않은 속성(Appearance, Behavior, Context)

- FDA의 조사 내용은 각 전자문서 포맷에 대한 SP 분석이기 때문에, 데이터세트의 SP에 맞춰서 Appearance와 Context 속성이 도출되지 않음
- Non-DB형 데이터세트는 전자문서와 달리 폰트, 레이아웃 등이 중요하지 않고, 데이터 및 기능이 외형보다 훨씬 보존 가치가 높기 때문에 Appearance 속성이 도출되지 않음.
- Non-DB형 데이터세트는 DB형처럼 SQL문을 통한 질의가 이루어지지 않고, VBA 매크로 인코딩 및 외부 링크, 외부 스키마 등은 부수적인 요소이기 때문에 Behavior 속성이 도출되지 않음
- 메타데이터와 관련된 Context는 전자문서와 데이터세트가 구별되는 특징이 없기 때문에 데이터세트의 고유기준으로 도출되지 않음
- 이러한 사항은 아래의 <표 52>과 같음

SP	전자기록	Non-DB형 데이터세트
Appearance	레이아웃, 폰트, 컬러 등 외형적인 요소들이 중요	중요하게 다루어지지 않음
Behavior	외부와의 연결이 거의 이루어지지 않기 때문에 중요하지 않음	DB형 데이터세트처럼 SQL문을 통해 외부와의 질의가 이루어지지 않음
Content	문자, 숫자 등 보존하기 어렵지 않은 것으로 이루어져 있기 때문에 비교적 중요하게 다루지 않음	데이터 및 수식이 중요
Context	제목, 작성자(생산자), 설명 메타데이터 등 내용 유사	
Structure	문자 인코딩, 템플릿, 스키마 등 사전 정의된 구조를 중요하게 다룸	중요하게 다루고 있으나, 특히 테이블을 구성하는 셀 위치(행, 열) 구조정보를 중요하게 다룸

<표 52> 전자기록과 Non-DB형 데이터세트의 SP 비교

○ Significant Properties와 Non-DB형 데이터세트 특징을 비교하여 Non-DB형 데이터세트의 특성 도출

- Non-DB형 데이터세트 특성을 도출하면 다음 <표 53>와 같음

Significant Properties	특성 설명	데이터세트의 특성
Content	· 데이터세트는 정형 데이터뿐만 아니라 전자문서 및 이미지와 같은 비정형 데이터, 여러 가지 데이터타입이 데이터세트 내에 포함되므로 Significant Properties의 Content와 매핑됨	이질성 (Heterogeneity)
Structure	· 현재 상용화된 응용프로그램을 통하여 다양한 유형의 데이터세트를 생산하는 성질이므로 Significant Properties의 Structure과 매핑되며, 응용프로그램은 업데이트를 거쳐 더욱 진화하기 때문에 여러 버전들이 생김 · 예) Excel 2010, Excel 2013, OpenOffice 4.1.5 등	다양성 (Diversity)

<표 53> Non-DB형 데이터세트 특성

○ 데이터세트 유형 전자기록 특성을 통한 선정기준 수립

- 앞서 분석한 내용을 바탕으로 데이터세트 특성에 대한 선정기준의 주요 항목은 <표 54>과 같음

- 수용성(Acceptability)의 경우, 아직 용어에 대한 검증이 이루어지지 않았으며, 다른 후보 용어로서 'Convertibility', 'Transferability', 'Acceptability' 등이 있음

- 복원성(Reproducibility)의 경우, 원래 용어인 '재현성'을 후보 용어인 'Stability', 'Rehabilitation', 'Reproducibility', 'Restoration' 중 'Reproducibility'를 채택하여 '복원성'으로 수정

- 활용성(Usability)의 경우, 기록의 4대 속성인 이용가능성(availability)와 용어가 비슷한 부분이 있으므로 논의 필요

데이터세트의 특성	선정기준	설명
이질성 (Heterogeneity)	수용성 (Acceptability)	· 데이터세트 내 데이터 타입(문자형, 숫자형, 문장형, 이진형), External File(비정형 데이터)를 보존 가능해야 한다는 기준
복잡성 (Complexity)	복원성 (Reproducibility)	· 데이터세트가 보존포맷으로 완벽히 변환 및 복원되어야 한다는 기준 · 수용성과 달리 Non-DB형 데이터세트 종류가 달라도 복원이 가능한 것이 중요
다양성 (Diversity)	호환성 (Compatibility)	· 보존포맷은 상용화된 다양한 종류의 Non-DB형 데이터세트와 호환이 가능해야 한다는 기준 · 보존포맷이 오픈소스일 경우, 지원하지 않은 Non-DB형 데이터세트를 호환 가능하게 하는 것이 중요

<표 54> Non-DB형 데이터세트 보존포맷 선정을 위한 기준항목 및 설명

○ 데이터세트 보존포맷 선정기준과 기록의 4대 속성과의 연관성 도출

- 도출된 선정기준과 기록의 4대 속성과의 연관성을 분석하면 다음과 같음

· Significant Properties는 진본성을 보장하는 특성 도출 틀이기 때문에 선정 기준은 진본성을 보장 가능함

· 수용성과 복원성의 경우, 업무 증거인 첨부파일 보존 또는 복원이 이루어져야 하므로 업무활동의 신뢰성을 보장할 수 있음

· 무결성은 보존포맷으로 변환되고, 복원 될 때, 무결한 상태에서 단계를 진행해야 하기 때문에 수용성, 복원성에서 강하게 관련됨

· 이용가능성은 보존이 제대로 이루어지지 않는다면 이용이 불가능하기 때문에 3가지 항목과 강하게 연관됨

나. 평가체계

○ 데이터세트 보존포맷 선정기준 기반 평가 체크리스트 개발(<표 55> 참고)

- 앞서 개발한 데이터세트[DB형] 유형 전자기록 보존포맷 선정기준에 따른 평가체계(안)을 차용함

평가 영역	평가 항목		Y/N
수용성	1	· 보존포맷은 데이터세트의 DataType(문자형, 숫자형, 날짜형, 이진형, 대용량)을 보존이 가능한가?	Y/N
	2	· 보존포맷은 데이터세트의 External File을 보존이 가능한가?	Y/N
활용성	3	· 보존포맷은 데이터세트의 DataType(문자형, 숫자형, 날짜형, 이진형, 대용량)을 복원이 가능한가?	Y/N
	4	· 보존포맷은 데이터세트의 External File을 복원이 가능한가?	Y/N
	5	· 보존포맷은 원래 생성된 Non-DB형 데이터세트 포맷이 아닌 다른 Non-DB형 데이터세트 포맷으로 복원이 가능한가?	Y/N
호환성	6	· 보존포맷은 4개 이상의 Non-DB형 데이터세트 포맷과 호환이 가능한가?	Y/N
	7	· 보존포맷은 3개 이상의 Non-DB형 데이터세트 포맷과 호환이 가능한가?	Y/N
	8	· 보존포맷은 1개 이상의 Non-DB형 데이터세트 포맷과 호환이 가능한가?	Y/N
	9	· 보존포맷이 오픈소스로 구성되어져 있는가?	Y/N

<표 55> Non-DB형 데이터세트 보존포맷 평가표

다. 포맷제시(예)

평가 영역		평가 항목	BIFF8	CSV	XML 1.0	XML DTD 1.0
수용성	1	· 보존포맷은 데이터세트의 DataType(문자형, 숫자형, 날짜형, 이진형, 대용량)을 보존이 가능한가?				
	2	· 보존포맷은 데이터세트의 External File을 보존이 가능한가?				
활용성	3	· 보존포맷은 데이터세트의 DataType(문자형, 숫자형, 날짜형, 이진형, 대용량)을 복원이 가능한가?				
	4	· 보존포맷은 데이터세트의 External File을 복원이 가능한가?				
	5	· 보존포맷은 원래 생성된 Non-DB형 데이터세트 포맷이 아닌 다른 Non-DB형 데이터세트 포맷으로 복원이 가능한가?				
호환성	6	· 보존포맷은 4개 이상의 Non-DB형 데이터세트 포맷과 호환이 가능한가?				
	7	· 보존포맷은 3개 이상의 Non-DB형 데이터세트 포맷과 호환이 가능한가?				
	8	· 보존포맷은 1개 이상의 Non-DB형 데이터세트 포맷과 호환이 가능한가?				
	9	· 보존포맷이 오픈소스로 구성되어져 있는가?				

<표 56> 각 Non-DB형 데이터세트 보존포맷 별 평가표(예시)

○ 포맷제시: 일반적으로 BIFF8, CSV가 Non-DB형 데이터세트 보존포맷으로 제시될 수 있으나, 이에 대해서는 추가적인 검토가 필요함

3. 국산 DBMS 큐브리드 대상 보존포맷 변환기능 개발

- 본 연구에서는 주관기관과 협의한 결과, 현재 G클라우드 표준으로 등록되어 있는 국내 유일의 DBMS인 큐브리드를 대상으로 SIARD Suite에 확장한 보존포맷 변환기능을 진행하여 개발을 완료함
- 3.1에서는 큐브리드 DBMS를 SIARD Suite에 확장 도입하기 위한 구현 방향을 소개함
- 3.2에서는 SIARD Suite을 구성하는 오픈소스코드 중 어떤 부분을 수정해야 하는지를 소개함
- 3.3에서는 실제 구현 과정과 구현한 소스코드에 대해 설명함

3.1 큐브리드 확장 SIARD Suite 오픈소스 구현 방향

○ 본 연구에서는 주관기관과의 협의에 따라 큐브리드 확장 SIARD Suite을 개발함

○ 큐브리드와 같이 SIARD Suite의 지원 DBMS 목록에는 포함하지 않는 새로운 DBMS를 SIARD Suite의 지원 DBMS 목록에 포함시키기 위해서는 해당 큐브리드 JDBC Driver 추가하고, SIARD Suite에 맞는 큐브리드 JDBC Wrapper를 제작해야 함

※ 큐브리드 JDBC Driver(cubrid_jdbc.jar)와 큐브리드 JDBC Wrapper(JdbcCubrid(jdbccubrid.jar))를 제작, 추가함

○ 지원 목록에 포함하는 것을 넘어서, **해당 DBMS에서 제공하는 모든 기능을 SIARD 파일로 Download하고 다시 해당 DBMS로 Upload**하기 위해서는, 가장 먼저 SIARD Suite에서 미지원하는 Type들을 분석하고, SIARD Suite을 구성하는 여러 프로젝트들의 소스코드에 Type 지원을 위한 부가적인 기능을 구현하여야 함

○ 본 연구에서는 큐브리드 DBMS를 대상으로 구현 방법 및 향후 다른 DBMS를 추가할 때의 구현 계획 및 방향 설정에 대해 설명함

3.1.1 큐브리드 DBMS Type 분류

○ 첫 번째, 큐브리드 제공 타입들은 (1) 기본 Data Type, (2) 특수 Data Type, (3) Key Type, (4) Routine Type으로 분류할 수 있음 (<표 57> 참고)

구분	큐브리드 Type
기본 Data Type	BIT, BIT VARYING, SHORT, INT, SMALLINT, BIGINT, NUMERIC, DECIMAL, FLOAT, REAL, DOUBLE, CHAR, VARCHAR, STRING, BLOB, CLOB, DATE, TIME, TIMESTAMP, DATETIME
특수 Data Type	SERIAL, Collection.SET, Collection.MULTISET, Collection.LIST, Collection.SEQUENCE, ENUM
Key Type	Primary Key, Foreign Key
Routine Type	Stored Procedure, Stored Function, Trigger

<표 57> 큐브리드 지원 요소 분류

3.1.2 요소 분류에 따라 SIARD Suite 구현 방향 설정

○ 두 번째, SIARD Suite에서 이들 요소들의 지원 가능한 요소들인지를 판단하기 위해서는 SIARD2.1 표준(SQL:2008)과 함께 SIARD Suite에서 DBMS 접속할 때 사용하는 표준 JDBC 인터페이스들의 상세 기능을 조사 및 분석해야 함

○ 실제 관련 사항들을 조사 및 분석한 결과, DBMS에서 제공하고 있는 다양한 요소들에 따라 다른 구현 방향이 존재함

○ 실제 구현 경험으로, DBMS에서 지원하고 있는 요소들을 <표 57>의 요소 분류에 따라 구분하고, SIARD Suite를 구성하고 있는 JDBC Driver와 JDBC Wrapper(큐브리드의 경우, 큐브리드 JDBC Driver와 큐브리드 JDBC Wrapper(JdbcCubrid)임)의 API들의 활용 및 수정 가능 정도에 따라 구현 방향을 결정하는 것이 가장 효율적일 것으로 판단됨

○ <표 58>는 큐브리드 기본 Data Type에 대한 구현 방향임

종류	기본 Data Type			
	큐브리드 9.3	SIARD2.1 (※매핑 가능 타입)	SIARD Suite 구현 난이도	구현 방향
숫자	BIT	VARCHAR	하	· 큐브리드 JDBC 및 JdbcCubrid에서 큐브리드의 기본 Data Type을 SIARD2.1의 해당 Data Type으로 단순 매핑
	BIT VARYING	VARCHAR	하	
	SHORT	SMALLINT	하	
	INTEGER	INTEGER	하	
	INT	INTEGER	하	
	SMALLINT	SMALLINT	하	
	BIGINT	BIGINT	하	
	NUMERIC	NUMERIC	하	
	DECIMAL	NUMERIC	하	
	FLOAT	FLOAT	하	
	REAL	FLOAT	하	
	DOUBLE	DOUBLE	하	
문자	CHAR	CHAR	하	
	VARCHAR	VARCHAR	하	
	STRING	VARCHAR	하	
	ENUM	VARCHAR	하	
	BLOB	BLOB	하	
	CLOB	CLOB	하	
기타	DATE	DATE	하	
	TIME	TIME	하	
	TIMESTAMP	TIMESTAMP	하	
	DATETIME	TIMESTAMP	하	

<표 58> 큐브리드 DBMS의 기본 Data Type 구현 방향

○ SQL:2008 표준에는 큐브리드에서 제공하고 있는 SERIAL, SET, MULTISET, LIST, SEQUENCE, ENUM 등 특수 Data Type들은 정의되어 있지 않기 때문에 단순 매핑 이외에 추가 작업이 필요함(<표 59> 참고)

○ <표 59>의 Collection타입(SET, MULTISET, LIST, SEQUENCE, ENUM)의 경우, 기본 Data Type 중에서 가장 관련 있는 Type으로 매핑하고 해당 Type에 대해 부가적으로 처리하기 위한 구현 작업이 추가됨

○ <표 59>의 SERIAL Type은 예외적인 경우로, 사용자 입장에서는 Data Type이지만 내부적으로는 Routine Type 형태로 처리되기 때문에 별도의 구현 작업이 필요함

특수 Data Type			
큐브리드 9.3	SIARD2.1 (표준 포함 여부)	SIARD Suite 구현 난이도	구현 방향
SERIAL	미포함	상	· 관련 System Table을 SIARD 파일에 포함되도록 큐브리드 JDBC Driver 및 JdbcCubrid 수정 · 관련 System table을 이용하여 CREATE SERIAL 구문 생성
Collection.SET	미포함	중	· VACHAR로 매핑되도록 큐브리드 JDBC Driver 및 JdbcCubrid 수정
Collection.MULTISET	미포함	중	
Collection.LIST	미포함	중	
Collection.SEQUENCE	미포함	중	
ENUM	미포함	중	

<표 59> CUBIRD DBMS의 특수 Data Type 구현 방향

○ <표 60>는 Key Type 요소에 대한 구현 방향으로, JdbcBase 인터페이스를 구현할 때 다른 JDBC Wrapper(예, JdbcMysql)를 참고하여 큐브리드 JDBC Wrapper(JdbcCubrid)를 구현하면 Key Type들은 쉽게 지원할 수 있음

Key Type			
큐브리드 9.3	SIARD2.1 (표준 포함 여부)	SIARD Suite 구현 난이도	구현 방향
Primary Key	포함	하	· 다른 DBMS JDBC Wrapper 단순 참고 및 매핑
Foreign Key	포함	하	

<표 60> CUBIRD DBMS의 Key Type 구현 방향

○ SIARD Suite은 SQL:2008 표준에 있는 모든 요소를 지원하는 것을 목표로 하고 있기 때문에 SQL:2008 표준에 없는 요소들은 지원하지 않지만, Routine Type처럼 SQL:2008에는 정의되어 있더라도 SIARD Suite에서 지원 못하거나 일부만 지원하는 Type도 있음

○ 특히, Stored Procedure/Function은 각 DBMS 별로 정의하는 구문도 다르고 지원하는 기능 및 구현 방법도 각기 달라 SIARD Suite을 확장하여 별도 구현하는 것이 필요함

※ 예를 들어, Oracle은 Stored Procedure/Function의 Body가 PL/SQL로 작성되어 있지만, 큐브리드의 경우 JAVA Class로 되어 있음

○ SIARD Suite은 Download된 SIARD 파일엔 Stored Procedure/Function의 Attribute(예를 들어, Stored Procedure/Function 이름)만 포함하고 있고, Body는 포함하지 않으며, Upload 시에는 이 Attribute조차 복원되지 않음

○ Stored Procedure/Function Body의 경우 큐브리드는 JAVA class 파일로 되어 있어 BLOB 데이터 형식으로 Archive에 별도 저장할 방법을 찾아야 하고, DB Upload 에도 포함되도록 SIARD Suite을 수정해야 함

※ Upload시, JAVA class 파일이 DB가 아닌 해당 컴퓨터의 파일시스템(C 또는 D 드라이브)에 저장되기 때문에 SIARD Suite을 local 컴퓨터에서 파일시스템에 쓰기 권한이 있는 상황을 가정함

○ Trigger는 Action이나 조건을 정의하는 내용이 각기 다르므로 실제 Download된 SIARD 파일에는 누락되어 있음

○ 그러나, 기본적인 Trigger의 Attribute는 SIARD에 구현이 되어 있고, Download된 SIARD 파일의 Metadata에 추가할 수 있는 여지를 가지고 있음

○ <표 61>은 Routine Type에 대한 구현 방향을 정리함

Routine Type				
큐브리드 9.3		SIARD2.1 (표준 포함 여부)	SIARD Suite 구현 난이도	대처 방안
Stored Procedure	Attribute	포함	중 (Download된 SIARD 파일에만 포함)	· Upload하여 DBMS에 복원할 때에도 포함되도록 SIARD Suite 수정
	Body	포함 (구체적 내용 없음)	상 (Download된 SIARD 파일에도 포함)	· JAVA class 파일을 BLOB으로 매핑하여 SIARD 파일에 포함되도록 SIARD Suite 수정/확장
Stored Function	Attribute	포함	중 (Download된 SIARD 파일에만 포함)	· Upload하여 DBMS에 복원할 때에도 포함되도록 SIARD Suite 수정
	Body	포함 (구체적 내용 없음)	상 (Download된 SIARD 파일에도 포함)	· JAVA class 파일을 BLOB으로 매핑하여 SIARD 파일에 포함되도록 SIARD Suite 수정/확장
Trigger	Attribute	포함 (구체적 내용 없음)	상 (Download/Upload 안됨)	· 큐브리드 JDBC와 SiardCmd에 getTriggers(...) 기능을 추가
	Body	포함 (구체적 내용 없음)	상 (Download/Upload 안됨)	· 관련 System Table을 이용하여 CREATE Trigger 구문 생성

<표 61> CUBIRD DBMS의 Routine Type 구현 방향

3.2 큐브리드 확장 SIARD Suite 오픈소스 수정 범위

○ SIARD Suite은 여러 개의 요소(Github에서는 'Repository'의 용어를 사용)로 구성되어 있으며 <그림 38>에는 그 요소들과 요소들간의 관계를 보여 주고 있음

○ 본 과제에서는 현재 SIARD에서 지원하고 있는 DBMS에 큐브리드 DBMS를 추가 지원하도록 개발하는 것을 목표로 하여, SIARD Suite의 여러 요소들의 소스코드를 수정하였음

○ 수정한 요소는 <그림 38>의 음영 표시된 3개 박스이며, 아래 4개 요소로 정리할 수 있음

(1) Cubrid 요소 (큐브리드 JDBC Driver)

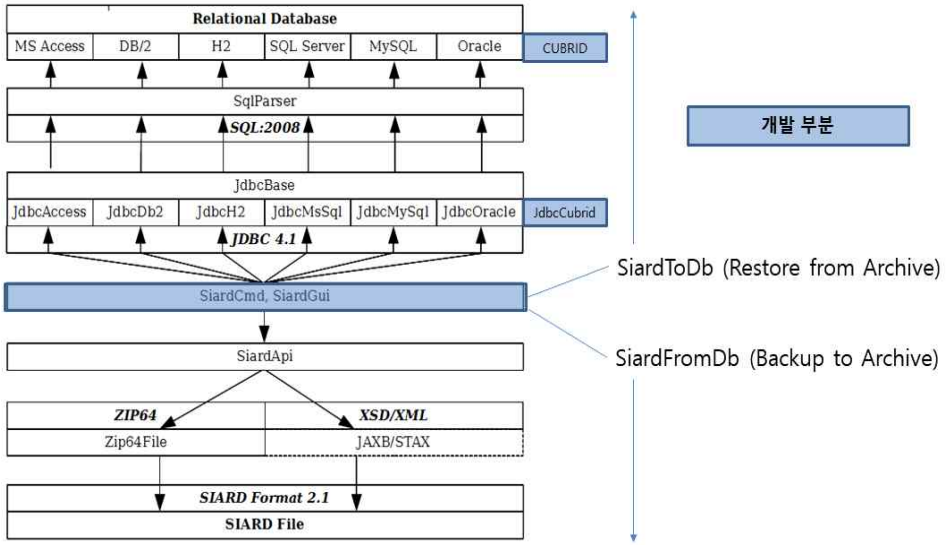
(2) JdbcCubrid 요소 (SIARD용 큐브리드 JDBC Wrapper)

(3) SiardCmd 요소 (SIARD command를 구현한 JAVA Application)

(4) SiardGui 요소 (SIARD GUI Application)

○ SIARD는 JDBC를 사용하여 DBMS에 접속하기 때문에 큐브리드 DBMS를 접속할 수 있는 큐브리드 JDBC Driver의 기능을 사용할 수 있어야 하고, 이를 위해서 큐브리드 JDBC Driver를 JdbcBase 자바 인터페이스(interface)로 감싸서(wrapping) 큐브리드 JDBC Wrapper인 JdbcCubrid 요소를 구현해야 함

○ 또한, <그림 38>과 같이 SiardGui는 내부적으로 SiardCmd와 SiardApi에 있는 기능들을 사용하는 구조로 되어있음. 즉, SiardCmd가 핵심 어플리케이션이자 라이브러리이며 SiardGui는 이를 활용한 GUI Application으로 볼 수 있음



<그림 38> 큐브리드 관련 Siard Suite 오픈소스 개발 범위

3.3 큐브리드 DBMS 확장 SIARD Suite 오픈소스 수정 내용

3.3.1 큐브리드 구현

○ SIARD Suite의 JdbcBase 인터페이스 즉, JDBC Wrapper를 구현하기 위해서 기존의 큐브리드 JDBC Driver를 수정하였으며, 수정 내용은 아래 <표 62>에 정리함

구분	수정 내용
API getTables(...) [CUBRIDDatabaseMetaData extends java.sql.DatabaseMetaData]	· getTables(...) JDBC API는 데이터베이스에 생성된 테이블 정보를 가져오는 API로 본래 "TABLE", "VIEW", "SYSTEM TABLE"을 입력 파라미터로 받아서 해당 정보를 ResultSet으로 반환하는 기능 수행 · SIARD 구현을 위해 큐브리드 JDBC Driver 루틴을 아래와 같이 변경함 ✓ 일반 TABLE정보에 _db_stored_procedure과 db_serial System Table 정보를 추가 ✓ Table 정보 중 Description에 "Show Create Table table-name"으로 생성되는 script text를 추가 (SIARD 데이터베이스 복원 시에 테이블 생성용으로 사용)
API toFile(...) [CUBRIDBlob extends java.sql.Blob]	· 큐브리드 데이터베이스는 Stored Procedure/Function Body가 JAVA JDBC 인터페이스 형태로 구현되어 있기 때문에, SIARD로 Archive할 때 Procedure/Function Body 부분을 BLOB 타입으로 저장하도록 JDBC Driver를 수정 · BLOB 데이터를 파일로 저장할 수 있도록 API 추가

API insertRow() [CUBRIDResultSet extends java.sql.ResultSet]	· SIARD데이터 복원 시 SERIAL 데이터를 처리하기 위해 db_serial 테이블에 데이터를 생성하는 시퀀스를 구현함
	· 일반적인 Insert into table-name 형태가 아닌 CREATE SERIAL ... 구문을 사용해야 하기 때문에 별도의 시퀀스를 도입
	`if (main_table_name.equalsIgnoreCase("db_serial")) {`
	`  if (updates[8] == null) {`
	`  sql = "create serial [" + updates[0] + "]" + " start with " + updates[2] + "`
	`  increment by " + updates[3] + " minvalue " + updates[5] + "`
	`  maxvalue " + updates[4];`
	`  if (updates[6].equals("0")) sql = sql + "cycle ";`
	`  if (!updates[10].equals("0")) sql = sql + "cache " + updates[10];`
	`  }`
	`  else return;`
	`}`
	· SIARD데이터 복원 시 Procedure/Function Body에 해당하는 JAVA Class 파일을 CUBRID JAVA 폴더에 저장하도록 코드 추가함

	`if (main_table_name.equalsIgnoreCase("_db_stored_procedure")) {`
	`  String filename = (String)updates[0];`
	`  String filepath = System.getenv("CUBRID") + "java" + File.separator;`
	`  File lOutFile = new File(filepath + filename);`
	`  FileOutputStream lFileOutputStream = new FileOutputStream(lOutFile);`
	`  lFileOutputStream.write((byte[])updates[1]);`
	`  lFileOutputStream.close();`
	`  return;`
	`}`

<표 62> 큐브리드 JDBC Driver 수정 내용

3.3.2 JdbcCubrid 구현

○ 기본적으로 SIARD는 JdbcBase Class로부터 JdbcMySql, JdbcOracle 등과 같은 Driver를 확장해서 원래 DBMS의 JDBC Driver로 연결되는 구조로 되어 있음

○ SIARD Suite의 핵심인 SiardCmd에서 이 확장 JDBC 인터페이스에서 제공하는 API를 호출하여 기능을 수행함

○ 본 과제에서는 우선 JdbcMySql를 기반으로 JdbcCubrid를 개발하는 형태로 진행하였으며, 수정 내용은 아래 <표 63>에 정리하였음

○ 기본적으로는 JdbcMySql를 그대로 가져와 구현하였으며, MySql과 동작 방식이 다른 API들은 큐브리드에 맞도록 수정하였음.

구분	수정 내용

CUBRIDDriver	· CUBRIDDriver에서는 BaseDriver를 등록하는 부분이 아래와 같이 수정되었음
	`public static void register()`
	`{`
	`  System.setProperty("jdbc.drivers", "cubrid.jdbc.driver.CUBRIDDriver");`
	`  try { BaseDriver.register(new CUBRIDDriver(),`
	`  "cubrid.jdbc.driver.CUBRIDDriver",`
	`  "jdbc:cubrid:localhost:33000:demodb:public::"); }`
	`  catch(Exception e) { throw new Error(e); }`
	`}`

CUBRIDConnection	· CUBRIDConnection에서 nativeSQL(String sql)의 경우 SIARD에서 수행되는 대부분의 SQL 구문을 파싱해서 SQL:2008 표준 형태로 재포맷을 하는 기능으로 큐브리드에서 지원하지 않는 형태의 구문에 대해서는 수정이 필요했으며 이를 수정하여 반영함 · 예를 들면, 큐브리드에서는 schema의 개념이 없는데, SIARD에서는 "schema-name"."table-name" 형태로 schema-name을 Qualifying하도록 재포맷을 하고 있어, 이를 schema-name을 제외하도록 수정하였음 · 큐브리드에서 schema-name은 database-name으로 매핑하여 사용

<table>
<tr>
<td>CUBRIDDatabaseMetaData</td>
<td>

· CUBRIDDatabaseMetaData에서는 아래 Meta 정보를 수집하기 위한 주요 기능이 포함되어 있으며, 이를 큐브리드에 맞도록 이들을 수정 또는 새로 작성함

```
getTables(), getColumns(), getTablePrivileges(), getColumnPrivileges(),
getPrimaryKeys(), getProcedures(), getFunctions(), getProcedureColumns(),
getFunctionColumns(), getAttributes(), getTriggers()
```

· 또한, 실제로 JDBC에는 정의되지 않은 getTriggers를 구현하였으며 이는, SIARD 2.1 스펙에 포함된 Trigger를 Archive에 포함시키기 위해 추가한 것임

· 현재의 SIARD 구현 스펙에는 Trigger가 포함되어 있지 않으며, 스펙 문서에만 포함되어 있는 상태로 JDBC에서 지원하지는 않지만 데이터베이스 보존에 꼭 필요한 요소이기 때문에 별도로 구현한 것임

· getProcedures에서 Procedure 관련 meta정보를 가져올 때 아래의 SQL 쿼리를 사용

```
String sSql = "SELECT₩r₩n" +"NULL as PROCEDURE_CAT,₩r₩n" +"'" +
            getConnection().getSchema() +
            "' as PROCEDURE_SCHEM,₩r₩n" +
            "SP_NAME AS PROCEDURE_NAME,₩r₩n" +
            "NULL as RESERVED1,₩r₩n" +
            "NULL as RESERVED2,₩r₩n" +
            "NULL as RESERVED3,₩r₩n" +
            "'AS Language ' + lang + ' NAME ' + '''' + target + '''' as
            REMARKS,₩r₩n" +
            "0 as PROCEDURE_TYPE,₩r₩n" +
            "NULL as SPECIFIC_NAME₩r₩n" +
            "FROM db_stored_procedure₩r₩n" +
            "WHERE " + sbCondition.toString() + "₩r₩n" +
            "ORDER BY PROCEDURE_CAT, PROCEDURE_SCHEM,
            PROCEDURE_NAME, SPECIFIC_NAME";
```

· 위 쿼리에서 Remarks에 세팅될 스트링은 아래의 CUBRID Stored Procedure 생성 구문의 AS 이하 뒷부분을 구현한 것임.

```
CREATE PROCEDURE procedure_name[(param[, param] ...)]
{IS | AS} LANGUAGE JAVA
NAME 'method_fullname (java_type_fullname[,java_type_fullname]...)
[return java_type_fullname]';

CREATE FUNCTION function_name[(param[, param]...)] RETURN sql_type
{IS | AS} LANGUAGE JAVA
NAME 'method_fullname (java_type_fullname[,java_type_fullname]...)
[return java_type_fullname]';
```

· 결국 위 코드를 응용하면 Stored Function의 생성 구문 또한 비슷하게 처리할 수 있음

</td>
</tr>
</table>

CUBRIDMetaColumns	· CUBRIDMetaColumns는 실제 테이블에 존재하는 Column이 아닌 일종의 가상 Column을 정의하는 기능으로 Stored Procedure나 Stored Function에서 파라미터를 정의할 때 사용되고 있으며, Column을 정의하기 위한 데이터 타입에 대해 아래 표와 같이 매핑하고 있음
	mapNAME_CUBRID_TO_ISO.put(CubridType.BIGINT, PreType.BIGINT);
	mapNAME_CUBRID_TO_ISO.put(CubridType.BIT, PreType.VARCHAR);
	mapNAME_CUBRID_TO_ISO.put(CubridType.BLOB, PreType.BLOB);
	mapNAME_CUBRID_TO_ISO.put(CubridType.CLOB, PreType.CLOB);
	mapNAME_CUBRID_TO_ISO.put(CubridType.CHAR, PreType.CHAR);
	mapNAME_CUBRID_TO_ISO.put(CubridType.DATE, PreType.DATE);
	mapNAME_CUBRID_TO_ISO.put(CubridType.DATETIME, PreType.TIMESTAMP);
	mapNAME_CUBRID_TO_ISO.put(CubridType.DECIMAL, PreType.DECIMAL);
	mapNAME_CUBRID_TO_ISO.put(CubridType.DOUBLE, PreType.DOUBLE);
	mapNAME_CUBRID_TO_ISO.put(CubridType.ENUM, PreType.VARCHAR);
	mapNAME_CUBRID_TO_ISO.put(CubridType.FLOAT, PreType.FLOAT);
	mapNAME_CUBRID_TO_ISO.put(CubridType.INT, PreType.INTEGER);
	mapNAME_CUBRID_TO_ISO.put(CubridType.INTEGER, PreType.INTEGER);
	mapNAME_CUBRID_TO_ISO.put(CubridType.SET, PreType.VARCHAR);
	mapNAME_CUBRID_TO_ISO.put(CubridType.SMALLINT, PreType.SMALLINT);
	mapNAME_CUBRID_TO_ISO.put(CubridType.TIME, PreType.TIME);
	mapNAME_CUBRID_TO_ISO.put(CubridType.TIMESTAMP, PreType.TIMESTAMP);
	mapNAME_CUBRID_TO_ISO.put(CubridType.STRING, PreType.VARCHAR);
	mapNAME_CUBRID_TO_ISO.put(CubridType.VARCHAR, PreType.VARCHAR);
	mapNAME_CUBRID_TO_ISO.put(CubridType.OBJECT, PreType.VARCHAR);

CUBRIDStatement	· CUBRIDStatement는 SIARD에서 발생하는 SQL를 처리하기 위한 Statement Driver로 executeQuery(...)가 주요 API이며 보통 SQL 구문 스트링을 nativeSQL로 재포맷하여 실제 JDBC Driver의 executeQuery(...)로 넘겨주게 되는데, 필요에 따라 큐브리드 문법에 맞는 SQL 구문 스트링을 변경하도록 구현함

<표 63> JdbcCubrid 수정 내용

3.3.3 SiardCmd 구현

가. SiardFromDb.java

○ SiardCmd에서 호출되는 SiardFromDb는 SIARD Archive를 생성하기 위한 JAVA코드로 크게 Meta 부분과 Primary 부분으로 나뉠 수 있음

○ Meta Data

- Meta 부분은 전체 Database의 스키마를 규정하기 위한 것으로 Archive 생성을 위해서는 MetaDataFromDb를 사용하고 있으며, 현재 대상 데이터베이스에 존재하는 데이블, 컬럼, 키/인덱스 정보, Stored Procedure/Function 정보, 트리거 정보 등을 구성 요소로 하는 xml 데이터를 생성함

○ 우선 테이블에 대해서는 JDBC의 getTables를 통해 테이블 관련 정보를 가져와 내부적으로 Meta Table 정보를 구성하고, 가져온 각 테이블에 대해 아래의 정보를 추가로 구성함

- 컬럼 정보 (getColumns)
- 키 정보 (getPrimaryKeys)
- 외래 키 정보 (getForeignKeys)
- 유일 키 정보 (getUniqueKeys)
- 트리거 정보 (getTirggers)
- 테이블의 레코드 개수 (getRows)

○ 위 정보 중 트리거 정보는 SIARD에서 누락되어 있던 것으로 새로 추가 구현하였으며, 아래의 그림은 getTables() 코드의 일부를 보여주고 있음

```
1329  private void getTables()
1330    throws IOException, SQLException
1331  {
1332    /* first count the tables for progress */
1333    String[] asTypes = new String[]{"TABLE"};
1334    if (_bViewsAsTables)
1335      asTypes = new String[]{"TABLE","VIEW"};
1336    ResultSet rs = _dmd.getTables(null, "%", "%", asTypes);
1337    _iTables = 0;
1338    while (rs.next())
1339      _iTables++;
1340    rs.close();
1341    _iTablesPercent = (_iTables+99)/100;
1342    _iTablesAnalyzed = 0;
1343    rs = _dmd.getTables(null, "%", "%", asTypes);
1344    while ((rs.next()) && (!cancelRequested()))
1345    {
1346      String sTableSchema = rs.getString("TABLE_SCHEM");
1347      String sTableName = rs.getString("TABLE_NAME");
1348      String sTableType = rs.getString("TABLE_TYPE");
1349      if (!Arrays.asList(asTypes).contains(sTableType))
1350        throw new IOException("Invalid table type found!");
1351      String sRemarks = rs.getString("REMARKS");
1352      Schema schema = _md.getArchive().getSchema(sTableSchema);
1353      if (schema == null)
1354        schema = _md.getArchive().createSchema(sTableSchema);
1355      Table table = schema.getTable(sTableName);
1356      if (table == null)
1357        table = schema.createTable(sTableName);
1358      MetaTable mt = table.getMetaTable();
1359      QualifiedId qiTable = new QualifiedId(null,sTableSchema,sTableName);
1360      System.out.println("  Table: "+qiTable.format());
1361      if ((sRemarks != null) && (sRemarks.length() > 0))
1362        mt.setDescription(sRemarks);
1363      getColumns(mt);
1364      getPrimaryKey(mt);
1365      getForeignKeys(mt);
1366      getUniqueKeys(mt);
1367      getTriggers(mt); /* Added for Trigger */
1368      getRows(mt);
1369      incTablesAnalyzed();
```

<그림 39> SIARD 포맷에 트리거 정보 입력을 위해 추가 코드

○ Primary Data

- Primary Data는 Meta 데이터 구성이 완료된 후 Archive에 실제 데이터를 채우는 것으로 PrimaryDataFromDb를 사용하고 있으며, download(...) API에서 출발해서, 스키마 내에 존재하는 각종 정보의 데이터를 데이터베이스로부터 가져와서 채우는 기능을 수행하는데

테이블이 그 중심이며 테이블을 기준으로 레코드를 가져오도록 구현되어 있음

- 큐브리드의 경우 stored procedure/function이 java class파일로 그 Body가 구현되어 있기 때문에, 테이블의 레코드를 가져올 때, procedure/function의 Body인 class파일을 Binary 형태로 가져오도록 추가 구현하였음

- 이를 위해 Meta Data를 구성할 때 실제 사용자 테이블이 아닌 _db_stored_procedure라는 일종의 시스템 테이블을 추가하였으며, 이 테이블 컬럼을 class filename과 procedure body로 구성하여 실제 procedure body에 대해서는 $CUBRID/java/*.class 파일을 읽어서 저장하도록 구현함

- 아래의 <그림 40>은 executeQuery(...)의 일부 코드를 캡처한 것으로 _db_stored_procedure를 포함하는 쿼리에 대해 SELECT #get#stored#body# 과 같이 일종의 메타 쿼리로 변경하여 실제 처리는 큐브리드 JDBC Driver에서 처리하도록 하였음

```
        CUBRIDResultSet rs;
        if (sNative.indexOf("FROM \"_db_stored_procedure\"") >=0) {
            if (sNative.indexOf("SELECT\r\n  \"sp_name\"") >= 0) {
                rs = new CUBRIDResultSet(super.executeQuery("SELECT #get#stored#body#"),_conn);
                rs.setPrimaryColumn(_conn.getTableWithoutPrimaryKey(), sPrimaryColumn);
                return rs;
            }

            if (sNative.indexOf("SELECT\r\n  COUNT(*)") >= 0) {
                rs = new CUBRIDResultSet(super.executeQuery("SELECT #get#stored#count#"),_conn);
                rs.setPrimaryColumn(_conn.getTableWithoutPrimaryKey(), sPrimaryColumn);
                return rs;
            }
        }

        rs = new CUBRIDResultSet(super.executeQuery(sNative),_conn);
        rs.setPrimaryColumn(_conn.getTableWithoutPrimaryKey(), sPrimaryColumn);
        return rs;
    } /* executeQuery */
```

<그림 40> executeQuery() 수정 코드

나. SiardToDb.java

○ SiardCmd에서 호출되는 SiardToDb는 SIARD Archive로부터 데이터베이스를 복원하기 위한 JAVA코드로 크게 Meta 부분과 Primary 부분으로 나뉠 수 있음

○ Meta Data

- Meta Data를 복원하기 위해서는 SiardCmd의 MetaDataToDb를 사용하며 세부적으로 아래의 기능들이 구현되었음

* createTables

* createColumn

* createProcedures

* createTirggers

- 위 리스트 중, createTriggers는 SIARD에 구현되어 있지 않은 것으로 추가로 구현되었다. SIARD는 SQL:2008 구문으로 재포맷하여 동작하는 구조로 되어 있으며, 이로 인해 DBMS 고유의 구문에 대해서는 예외 오류를 발생하는 경우가 종종 발생함

- 이를 처리하기 위해 큐브리드에서 제공되는 Show Create Table table-name 과 같은 구문을 이용하도록 일부 코드를 수정하였음

- 특히, Trigger의 큐브리드의 시스템 테이블인 db_trigger를 이용하였으며, 시리얼(SERIAL)에 대해서도 db_serial 테이블을 이용하여 구현하였음

- 아래의 <그림 41>은 createTriggers를 구현한 JAVA 코드 일부이다.

```
552   /*-----------------------------------------------------------------*/
553   /** create all triggers of a schema.
554    * @param ms schema meta data.
555    * @param sm mapping of names in schema.
556    * @throws IOException if an I/O error occurred.
557    * @throws SQLException if a database error occurred.
558    */
559   private void createTriggers(MetaTable mt)
560     throws IOException, SQLException
561   {
562     _il.enter(mt.getName());
563
564
565     for (int iTrigger = 0; iTrigger < mt.getMetaTriggers(); iTrigger++)
566     {
567       MetaTrigger mtr = mt.getMetaTrigger(iTrigger);
568
569       StringBuilder sbSql = new StringBuilder("");
570       sbSql.append(mtr.getDescription());
571
572       /* now execute it */
573       _il.event(sbSql.toString());
574       Statement stmt = _dmd.getConnection().createStatement();
575       stmt.setQueryTimeout(_iQueryTimeoutSeconds);
576       stmt.executeUpdate(sbSql.toString());
577       stmt.close();
578     }
579     _il.exit();
580   } /* createTriggers */
```

<그림 41> createTriggers() 구현 코드

○ Primary Data

- Primary Data의 경우 PrimaryDataToDb를 사용하며 JDBC의 Updatable ResultSet을 기반으로 insertRow 기능을 통해 데이터베이스에 실제 데이터를 추가하는 방식으로 구현되었음

- 실제 테이블이 아닌 _db_stored_procedure나 db_trigger, db_serial의 경우 각각 예외 처리를 통해 실제 데이터를 복원하는 방식으로 구현하였음

- 아래 <그림 42>의 ResultSet의 JDBC insertRow 코드의 일부로 db_serial의 경우 db_serail 테이블에 Row를 추가하는 것이 아니라 Serial을 CREATE하는 구문으로 만들어서 실행을 하는 형태로 구현하였음

```
1322    public synchronized void insertRow() throws SQLException {
1323        try {
1324            synchronized (con) {
1325                synchronized (this) {
1326                    checkIsOpen();
1327                    if (main_table_name.equalsIgnoreCase("db_serial")
1328                      ||main_table_name.equalsIgnoreCase("_db_stored_procedure")
1329                    ) {
1330                        is_updatable = true;
1331                    }
1332                    else {
1333                        checkIsUpdatable();
1334                        if (!inserting) {
1335                            throw con.createCUBRIDException(
1336                                    CUBRIDJDBCErrorCode.invalid_row, null);
1337                        }
1338                    }
1339                    if (main_table_name == null) {
1340                        return;
1341                    }
1342
1343                    String sql = null;
1344                    if (main_table_name.equalsIgnoreCase("db_serial")) {
1345                        if (updates[8] == null) {
1346                            sql = "create serial [" + updates[0] + "]"
1347                                    + " start with " + updates[2]
1348                                    + " increment by " + updates[3]
1349                                    + " minvalue " + updates[5]
1350                                    + " maxvalue " + updates[4];
1351                            if (updates[6].equals("0")) sql = sql + "cycle ";
1352                            if (!updates[10].equals("0")) sql = sql + "cache " + updates[10];
1353                        }
1354                        else return;
1355                    }
```

<그림 42> JDBC insertRow() 수정 코드

다. MetaDataFromDb.java

○ SIARD2.1 표준에는 정의되어 있으나 SIARD Suite에는 실제로 구현되지 않은 Trigger, Stored Procedure 등을 DB로부터 Download하기 위해 getTriggers(), getRoutines(), getTables()를 수정 및 새로 작성함

구분	수정 내용
getRoutines(...)	· Stored Procedure에 대해 Remarks를 Description에 추가하고 setCharacteristic에 "PROCEDURE"를 세팅함 · Remarks는 getStoredProcedures에서 Procedure를 정의하는 부분이 세팅되어 옴
getTriggers(...)	· 원래 Siard에는 Trigger 관련 코드가 누락되어 있기 때문에, 큐브리드에서 사용하는 Trigger Syntax에 맞게 아래의 코드를 추가 · Trigger를 가져오는 방법으로 db_trigger 시스템 테이블을 사용하였으며, db_trigger에 포함된 내용을 SIARD의 xml에 정의된 Attribute로 가공해야 함 (SQL구문에 포함)

<table>
<tr><td></td><td>

```
private getTriggers(MetaTable mt)
{
...
    String sSql = "SELECT₩n" + "name,₩n" +
    "decode(condition_type,null,decode(action_time,1,'BEFORE',2,'AFTER',3,'INSTE
    AD'), decode(condition_time, null, 'AFTER', 1, 'BEFORE', 2, 'AFTER', 3,
    'INSTEAD')) as action_time,₩n" +"'EXECUTE '||
    decode(condition_type,null,'',decode(action_time,1,'',2,'AFTER ',3,'DEFERRED
    '))|| decode(action_type,1,action_definition,2,'reject',3,'invalidate
    transaction',4,'PRINT ''' || action_definition||''''') as triggered_action,₩n"
    +"decode(event,0,'UPDATE',1, 'STATEMENT
    UPDATE',2,'DELETE',3,'STATEMENT DELETE',4,'INSERT',5,'STATEMENT
    INSERT',8,'COMMIT',9,'ROLLBACK') || " +"' ON ['||(select target_class_name
    from db_trig where trigger_name=name)|| ']' ||₩n" +"
    decode(target_attribute,null,'',' (['||target_attribute||'])') as trigger_event,₩n"
    +" (select 'CREATE TRIGGER '||'[' || name || ']' ||₩n" + "            ' STATUS
    '|| decode(status,1,'INACTIVE',2,'ACTIVE') ||₩n" + "            ' PRIORITY
    '||cast(PRIORITY as numeric(10,5)) ||' ' ||₩n" + "
    decode(condition_type,null,decode(action_time,1,'BEFORE ',2,'AFTER
    ',3,'DEFERRED '), decode(condition_time, null, 'AFTER', 1, 'BEFORE', 2,
    'AFTER', 3, 'DEFERRED')) || ' ' ||₩n" + "
    decode(event,0,'UPDATE',1, 'STATEMENT
    UPDATE',2,'DELETE',3,'STATEMENT DELETE',4,'INSERT',5,'STATEMENT
    INSERT',8,'COMMIT',9,'ROLLBACK') ||₩n" + "                    ' ON ['||(select
    target_class_name from db_trig where trigger_name=name)|| ']' ||₩n" + "
            decode(target_attribute,null,'',' (['||target_attribute||'])') ||₩n" + "
        decode(condition,null,'',' if '||condition) ||₩n" + "          ' EXECUTE
    '|| decode(condition_type,null,'',decode(action_time,1,'',2,'AFTER
    ',3,'DEFERRED '))||
    decode(action_type,1,action_definition,2,'reject',3,'invalidate
    transaction',4,'PRINT ''' || action_definition)₩n" + "from db_trigger b₩n" +
    "where a.name = b.name) as description₩n" + "FROM db_trigger a₩n"
    +"WHERE " + sbCondition.toString();
...
}
```

</td></tr>
<tr><td>getTables()</td><td>

· getTables는 DB 테이블 관련 모든 내용을 가져오는 루틴으로 MetaDataFromDb에 정의되어 있음

· 아래와 같이 표와 같은 getTriggers(..)를 추가

```
...
getColumns(mt);
getPrimaryKey(mt);
getForeignKeys(mt);
getUniqueKeys(mt);
if (_cubrid) { → 추가된 부분 [CUBRID에서만 동작하도록]
    getTriggers(mt); /* Added for Trigger */
}
getRows(mt);
incTablesAnalyzed();
...
```

</td></tr>
</table>

<표 64> MetaDataFromDb.java 수정 내용

마. MetaDataToDb.java

○ 타 DBMS에서 생성된 SIARD 파일을 복원할 때, 중요한 내용 중 하나가 타입 변환인데 타입 자체 매핑과 타입에서 사용할 수 있는 제약 조건을 고려하여 복원하는 부분을 수정하였으며, 수정 내용을 <표 65>에 정리함

구분	수정 내용
createColumn(...)	· 제약 조건 중 가장 큰 부분은 자릿수인데 예를 들면, 수치형 타입의 최댓값이 DBMS별로 다르기 때문에 이 부분을 고려하여 처리해야 함 · 큐브리드의 경우 수치형 자릿수는 38자리이고 문자형 자릿수는 1기가(1,073,741,823)로 제한되어 있어서 타 DBMS의 데이터를 큐브리드로 복원할 때 이를 반영해야 함. · 또한 BINARY 타입은 큐브리드에서 지원하지 않으므로, 이를 복원할 때 BIT혹은 BIT VARYING으로 매핑하여 복원해야 함. 다만 ORACLE의 BINARY데이터 길이는 2기가까지 가능하지만 큐브리드의 BIT타입은 1기가까지 가능하고 BINARY데이터는 byte단위의 길이이고 BIT타입은 bit단위의 길이로 BINARY타입의 자릿수를 체크할 때 1/8로 계산해야 함
createTables(...)	· SIARD 파일이 큐브리드 DBMS에서 Download된 파일일 때, db_serial과 _db_stored_procedure 테이블은 실제로 만들지 않도록 조치 · 큐브리드 SIARD 파일 → 큐브리드 DB로 복원하는 경우에만 createCubridTable(...)을 이용 · 또한, 큐브리드 SIARD 파일 → 큐브리드 DB로 복원하는 경우에만 createTriggers(..)를 이용
createProcedures(...)	· createProcedures(...)는 큐브리드로 복원할 때만 사용, 물론 큐브리드 archive를 사용할 때만 적용

<표 65> MetaDataToDb.java 수정 내용

바. PrimaryDataToDb.java

○ putTable과 pubSchema는 SIARD 파일에 저장된 테이블을 복원하는 것으로 SIARD 파일이 큐브리드에서 Download하여 만들어진 경우 _db_stored_procedure와 db_serial은 복원하지 않음

○ 테이블을 복원하는 것이 아니라, 실제 데이터를 복원할 때 createProcedure(...)를 통해 stored procedure/function을 복원하고, 실제 db_serial 데이터를 복원할 때 db_serial 테이블을 만들지 않고, CREATE SERIAL... 구문으로 대체하여 SEIRAL 데이터를 생성

구분	수정 내용

putSchema(...)	String todb = _conn.getMetaData().getDatabaseProductName();
	String ardb = _archive.getMetaData().getDatabaseProduct().substring(0,6);
	for (int iTable = 0; (iTable < schema.getTables()) &&
	(!cancelRequested()); iTable++)
	{
	Table table = schema.getTable(iTable);
	if (ardb.equals("CUBRID") && !todb.equals("CUBRID")) {
	MetaTable mt = table.getMetaTable();
	if (mt.getName().equals("_db_stored_procedure")) continue;
	if (mt.getName().equals("db_serial")) continue;
	}
	putTable(table,sm);
	}

<표 66> putSchema(...) 수정내용

3.3.4 SiardGui 구현

○ Trigger 타입을 SiardGui 윈도우에서 보여 주기 위한 소스코드 수정

- SIARD 2.1 표준에는 Trigger 타입에 대한 내용이 있지만, SIARD Suite 에서는 Trigger 타입을 DBMS으로부터 Download하지 않기 때문에, 큐브리드 Trigger 타입을 Download 할 수 있는 기능을 SiardCmd에 추가함. 이를 위해 Download된 Trigger 타입을 SiardGui 윈도우 화면에서 보여 주기 위해 소스 코드를 수정함(<표 67> 참고)

구분	수정 내용

MetaDataTableFactory.
java
	public static ObjectListTableView newMetaTriggersTableView(MetaTable mt)
	{
	SiardBundle sb = SiardBundle.getSiardBundle();
	List<String> listHeaders = Arrays.asList(
	sb.getProperty("header.metatriggers.row"),
	sb.getProperty("header.metatriggers.name"),
	sb.getProperty("header.metatriggers.actiontime"),
	sb.getProperty("header.metatriggers.triggeredevent"),
	sb.getProperty("header.metatriggers.triggeraction"));
	ObjectListTableView oltv = new ObjectListTableView(listHeaders);
	for (int iTrigger = 0; iTrigger < mt.getMetaTriggers(); iTrigger++)
	{
	MetaTrigger mtr = mt.getMetaTrigger(iTrigger);
	List<Object> listRow = Arrays.asList((Object)iTrigger,
	mtr.getName(),mtr.getActionTime(),mtr.getTriggerEvent(),mtr.getTriggeredAction());
	oltv.getItems().add(listRow);
	}
	return oltv;
	} /* newMetaForeignKeysTableView */

ArchiveTreeView.java	

```
private class MetaTriggerTreeItem extends DynamicTreeItem<Object>
{
  public MetaTriggerTreeItem(MetaTrigger mtr)
  {
    super(mtr);
  } /* constructor */
  @Override
  public void addChildren()
  {
    // foreign key has no children
  } /* addChildren */
} /* class MetaForeignKeyTreeItem */

private class MetaTriggersTreeItem extends DynamicTreeItem<Object>
{
  public MetaTriggersTreeItem(MetaTable mt)
  {
    super(mt);
  } /* constructor */
  @Override
  protected void addChildren()
  {
    MetaTable mt = (MetaTable)getValue();
    for (int iMetaTrigger = 0; iMetaTrigger < mt.getMetaTriggers(); iMetaTrigger++)
        getChildren().add(new MetaTriggerTreeItem(mt.getMetaTrigger(iMetaTrigger)));
  } /* addChildren */
} /* class MetaTriggersTreeItem */
```

※ 수정된 모든 코드 아닌 주요 코드를 발췌함

<표 67> SiardGui 수정 내용

3.4 오픈소스 라이선스 관련 소스 공개 범위

○ 큐브리드 확장 SIARD Suite에서 수정한 개발한 SIARD Suite 프로젝트 요소(Repository)는 아래와 같음

- Cubrid(큐브리드 JDBC driver)
- JdbcCubrid(큐브리드 JDBC Wrapper)
- SiardCmd(SIARD command를 구현한 JAVA 어플리케이션)
- SiardGui(SIARD GUI 어플리케이션)

○ 위 코드 중 공개해야 할 코드, cubrid JDBC driver, SiardCmd, SiardGui이며, **JdbcCubrid는 공개하지 않아도 됨**

4. 데이터세트 유형 전자기록 보존포맷 변환 · 복원 검증

- 데이터세트 유형 전자기록 보존포맷(SIARD)의 적합성 여부를 판단하기 위해 검증 시험을 진행하였음
- SIARD 변환 대상 DBMS는 기관 협의를 통해 4종(MySQL, SQL Server, Oracle, 큐브리드)의 DBMS를 선정
- 기존의 SIARD는 국산 DBMS인 큐브리드를 지원하지 않기 때문에 SW 개발을 통해 큐브리드를 지원하는 버전의 SIARD를 사용하여 검증 시험을 진행함(SW 개발과 관련된 내용은 5장을 참고)
- 4종의 DBMS ↔ SIARD 변환 시험을 "사전 시험", "본 시험", "DB Size 시험"으로 나눠서 진행
- "사전 시험"은 SIARD에서 제공하는 기본적인 변환 기능에 대한 검증을 목적으로 진행하였으며, 사전 시험을 통해 4종 DBMS의 Data Type, Key Type, Routine Type 중 SIARD 포맷으로 변환이 가능한 부분과 불가능한 부분을 도출(결과요약은 <표 68> 참고)
- "본 시험"은 DBMS의 Data와 SIARD 포맷으로 변환된 Data들에 대해 원본DB와 업로드DB의 동일 여부 검증하였으며, 변환 검증 시험의 Data 동일 여부에 관한 검증은 DBMS 종류에 따라 TOAD Data Point, 큐브리드 MANAGER에서 제공하는 비교 마법사 TOOL을 사용하여 검증을 진행하였음(결과요약은 <표 77> 참고)
- "DB Size 시험"은 DB SIZE 별도 Download 및 Upload의 시간 및 변환 여부 검증(결과요약은 <표 70> 참고)

항목 \ DBMS	MySQL	SQL Server	Oracle	큐브리드	
				SW개발 전	SW개발 후
일반 Data Type	◎	◎	◎	X	◎
특수 Data Type	○	◎	○	X	◎
Key Type (PK, FK)	◎	◎	○	X	◎
Routine Type	X	X	X	X	◎

<표 68> "사전 시험" : 4종 DBMS↔SIARD Type별 변환 가능 검증 시험 결과 요약표

(◎: 모두 변환 가능, O: 부분 변환 가능, X: 변환 불가능)

항목 \ DBMS	MySQL	SQL Server	Oracle	큐브리드
Data	◎	◎	◎	◎

<표 69> “본 시험” : 4종 DBMS↔SIARD 변환 · 복원 Data 동일 검증 결과 요약표

DB Size	table 개수	총 레코드 개수	Download 소요시간	Upload 소요시간
약 1GB (1,112MB)	5개	총 331,500개	약 20분	약 2분
약 2GB (2.205MB)	5개	총 662,500개	약 40분	약 4분
약 3GB (3,416MB)	5개	총 823,500개	약 1시간	약 6분
약 4GB (4,262MB)	5개	총 994,500개	약 1시간 15분	약 9분
약 5GB (5,300MB)	5개	총 1,123,500개	약 1시간 40분	약 13분
약 6GB (6,424MB)	5개	총 1,476,500개	약 2시간	약 16분
약 7GB (7,451MB)	5개	총 1,861,500개	약 2시간 30분	약 20분
약 8GB (9,046MB)	5개	총 2,264,500개	약 2시간 50분	약 28분
약 9GB (10,125MB)	5개	총 2,682,500개	약 3시간 15분	약 35분
약 10GB (11,864MB)	5개	총 3,014,500개	약 3시간 35분	약 40분

<표 70> “DB Size 시험” : DB Size별 Download, Upload 소요시간 요약표

○ 총 레코드의 개수가 늘어날수록 Download와 Upload하는데 소요되는 시간은 선형적으로 증가하며 Download 시간이 Upload 시간보다 9배 정도 더 걸림

○ 그러나, 같은 10GB라도 Table 개수, 레코드 개수, 컬럼 개수에 따라 Download와 Upload 속도는 달라질 수 있으며, Upload 시간이 더 많이 소요되는 경우도 많음

4.1 보존포맷 변환 검증 사전 시험 개요

- 보존포맷 변환 검증 사전 시험은 SIARD에서 제공하는 기본적인 기능 검증을 목적으로 진행
- 기존 SIARD는 큐브리드를 지원하지 않기 때문에, SW 개발을 하여 큐브리드를 지원하는 버전의 SIARD를 사용함
- 변환 검증 대상 4종 DBMS의 일반 및 특수 Data Type, Key Type, Routine Type의 변환 여부를 검증

○ 보존포맷 변환 검증 사전 시험에서 사용되는 용어 소개

- 본 시험에서 사용되는 용어를 소개함과 동시에 일관성 있는 용어 사용으로 인해 상호 간 명확한 의사소통으로 효율적인 연구 과제 진행을 기대함

용어	내용
Table	· 데이터베이스에서 정보를 구분하여 저장하는 기본 단위 · Row, Column 등을 포함
Row	· Table의 행으로, 값들의 나열(리스트) · 동의어 “Record”
Column	· Table의 열 · 동의어 “Attribute”
Cell	· Table의 행과 열이 교차하는 부분
Primary Key	· Primary Key(PK:기본 키)는 Table 내 Row를 구분하는데 기준이 되는 하나 혹은 그 이상의 Column의 집합인 후보 키 중 하나를 선정해 대표로 삼는 키
Foreign Key	· Foreign Key(FK: 외래 키)는 다른 Table의 PK를 참조하는 것으로 Table의 관계를 나타내기 위해 사용하는 키
Download	· SIARD Suite의 기능을 제공하는 DBMS의 특정 Schema, Tablespace, DB 등을 SIARD 파일로 변환하는 것
Upload	· 이용자가 SIARD Suite을 이용해 생성한 SIARD 파일을 DBMS의 DB로 변환하는 것
Open	· SIARD Suite을 이용해 SIARD 파일의 data, column, user 등의 정보를 확인할 수 있도록 하는 것
Close	· SIARD Suite에 Open 해놓은 SIARD 파일을 닫는 것
Metadata	· SIARD 파일, schema, table, column 등에 대한 메타데이터

<표 71> 보존포맷 변환 검증 사전 시험에서 사용되는 용어>

4.1.1 보존포맷 변환 검증 사전 시험 목적

○ 보존포맷 변환 검증 사전 시험 목적

- 보존포맷 변환 검증 사전 시험은 여러 항목들의 변환 가능 여부와 같은 SIARD의 기본적인 기능 확인에 초점을 맞춰 진행함
- 보존포맷 변환 검증 사전 시험은 내부 변환 검증 대상인 4종의 DBMS(MySQL, SQL Server, Oracle, 큐브리드를 SIARD 포맷으로 변환 가능 여부를 알아보는 것이 목적
- 4종 DBMS↔SIARD 포맷으로 정상 변환이 가능한 Data Type과 불가능한 Data Type 도출, Key Type, Routine Type 변환 가능 여부를 알아보는 것이 목적

4.1.2 보존포맷 변환 검증 사전 시험 방법 및 대상

○ 보존포맷 변환 검증 사전 시험 방법

- 보존포맷 변환 검증 사전 시험은 아래의 <표 72>과 같이 3단계로 진행

순서	상세 내용
1. DB 생성	· 4종의 DBMS에서 각각 DB 생성 · DB 생성 시 Routine Type도 포함하여 생성
2. SIARD 파일 생성	· 생성한 DB를 SIARD 파일로 변환
3. 동일한 DBMS로 Upload	· SIARD 파일을 본래의 DBMS로 Upload 하여 Data Type, Routine Type 보존 여부 확인

<표 72> 보존포맷 변환 검증 사전 시험 방법

○ 보존포맷 변환 검증 사전 시험 대상

- 4종의 DBMS는 DB의 구조와 사용하는 Data Type이 다르기 때문에, DBMS 제조사에서 제공하는 매뉴얼을 참고하여 최대한 많고 다양한 Data Type을 이용해 DB 생성
- 보존포맷 변환 검증 사전 시험에 사용한 4종 DBMS의 Data Type은 아래의 <표 73>과 같음

종류	Data Type			
	MySQL 8.0	SQL Server 2017	Oracle 11g	큐브리드 9.3
숫자	BIT	BIT	NUMBER	BIT
	INT	INT	FLOAT	BIT VARYING
	TINYINT	TINYINT	BINARY_FLOAT	INT
	SMALLINT	SMALLINT	BINARY_DOUBLE	SMALLINT
	MEDIUMINT	BIGINT		BIGINT
	BIGINT	MONEY		NUMERIC
	NUMERIC	SMALLMONEY		DECIMAL
	DECIMAL	NUMERIC		FLOAT
	DOUBLE	DECIMAL		DOUBLE
	REAL	FLOAT		
	FLOAT	REAL		
	BOOLEAN			
문자/이진	CHAR	CHAR	CHAR	CHAR
	VARCHAR	NCHAR	VARCHAR2	VARCHAR
	BINARY	VARCHAR	NCHAR	STRING
	VARBINARY	NVARCHAR	NVARCHAR2	
		BINARY		
		VARBINARY		
Large Object	BLOB	TEXT	LONG	BLOB
	TINYBLOB	NTEXT	RAW	CLOB
	MEDIUMBLOB	IMAGE	LONG RAW	
	LONGBLOB		BLOB	
	TEXT		BFILE	
	TINYTEXT		CLOB	
	MEDIUMTEXT		NCLOB	
	LONGTEXT			
	ENUM			ENUM
	SET			
날짜, 시간	DATE	DATE	DATE	DATE
	TIME	TIME	TIMESTAMP	TIME
	DATETIME	DATETIME	TIMESTAMP WITH TIME ZONE	TIMESTAMP
	TIMESTAMP	DATETIME2	TIMESTAMP WITH LOCAL TIME ZONE	DATETIME
	YEAR	DATETIMEOFFSET	INTERVAL YEAR TO MONTH	
		SMALLDATETIME	INTERVAL DAY TO SECOND	
기타	JSON	geography	ROWID	SET
	GEOMETRY	geometry	UROWID	MULTISET
	POINT			LIST, SEQUENCE
	MULTIPOINT			
	LINESTRING			
	MULTILINESTRING			
	POLYGON			
	MULTIPOLYGON			
	GEOMETRY COLLECTION			

<표 73> 4종 DBMS Data Type

4.1.3 보존포맷 변환 검증 사전 시험 환경 구축

○ 보존포맷 변환 검증 사전 시험 환경 구축

- 효율적인 보존포맷 변환 검증 사전 시험과 더불어 실데이터 검증을 대비하여 노트북을 이용해 시험 환경 구축

- 구축한 보존포맷 변환 검증 사전 시험을 위해 구축한 환경의 정보는 아래의 <표 74>와 같음

제조사	CPU	RAM	SSD	HDD
HP	i7-8750H 2.2GHz	32GB	1TB	1TB

<표 74> 보존포맷 변환 검증 시험 환경 정보

○ SIARD Suite Build 및 사용법 ※전체 내용은 [별첨03] 참고

- 기존의 SIARD Suite은 외산 DBMS만 지원하고, 국산 DBMS인 큐브리드는 지원하지 않음

- 따라서 본 시험에서는 기존의 SIARD Suite에 큐브리드를 추가로 지원하는 확장된 버전을 개발하여 진행함

- SIARD Suite Build 순서는 다음과 같음

1. SIARD Suite Build에 필요한 사전 SW 설치

① JDK(JAVA Development Kit) Version 1.8.0.201 다운로드 및 설치

② Eclipse IDE for JAVA EE Developers 설치

③ GIT 설치 및 실행

2. SIARD Suite Extension for 큐브리드 소스코드 다운로드

① Github 접속(Username: nak2019)

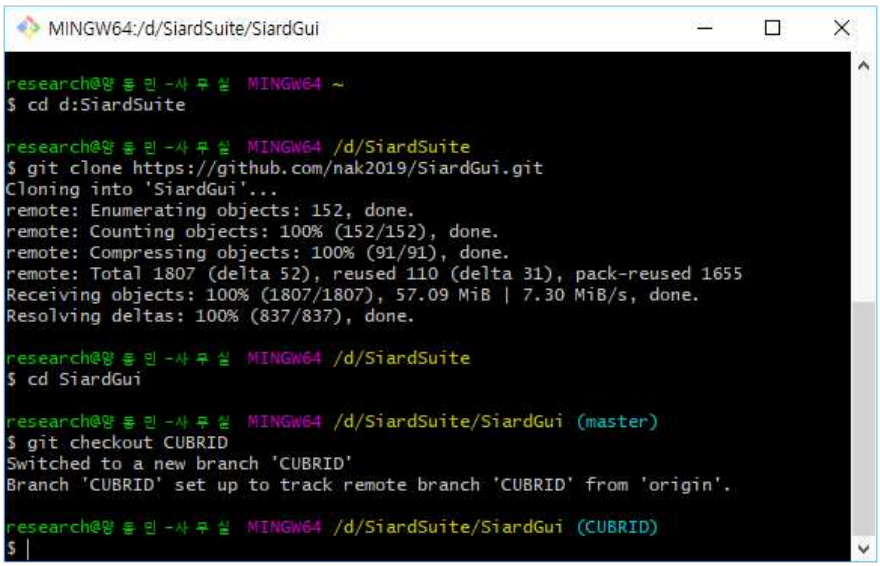
② Repository를 다운로드 할 폴더 생성(본 보고서에서는 “D:\\SiardSuite”으로 생성)

③ git bash 실행, “D:\\SiardSuite”으로 이동(cd d:SiardSuite 입력)

④ git clone + Download URL 구문을 활용해 4개의 Repository(SiardGui, SiardCmd, Cubrid, JdbcCubrid)(<그림 43>참고)

⑤ “D:\\SiardSuite\SiardGui”로 이동(cd d:SiardSuite:SiardGui 입력)

⑦ git checkout CUBRID 입력



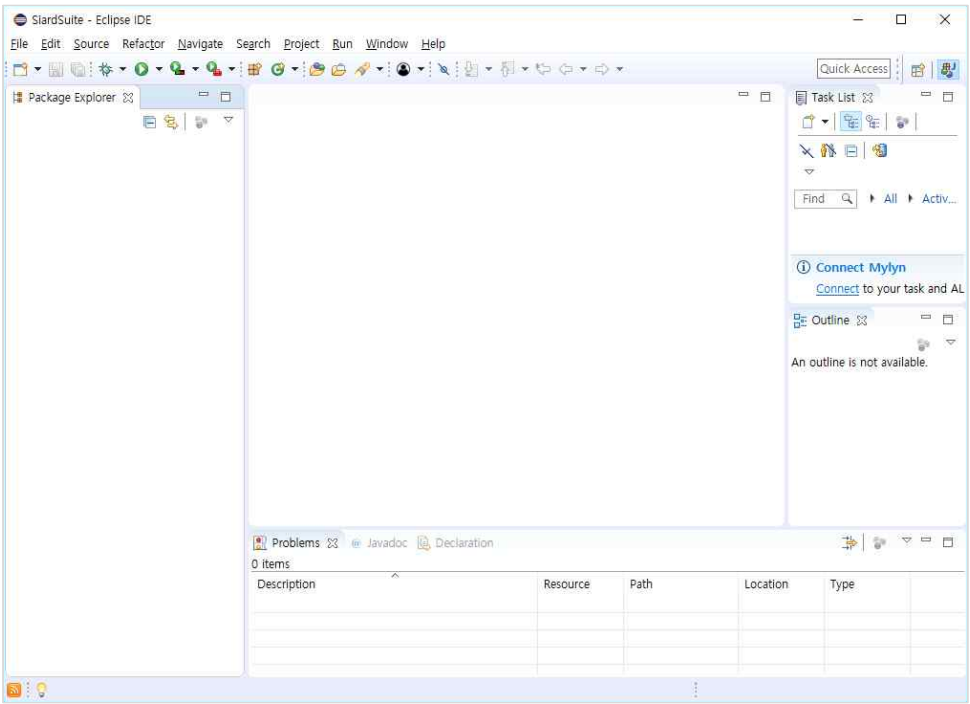
<그림 43> git으로 SiardGui 다운로드 방법

⑧ “D:\\SiardSuite\SiardCmd”로 이동(cd .. -> cd SiardCmd 입력)

⑨ git checkout CUBRID 입력

3. 4개의 Repository Eclipse Project로 추가

① Eclipse 실행 → Workspace “D:\SiardSuite”으로 설정



<그림 44> Eclipse 실행 첫 화면

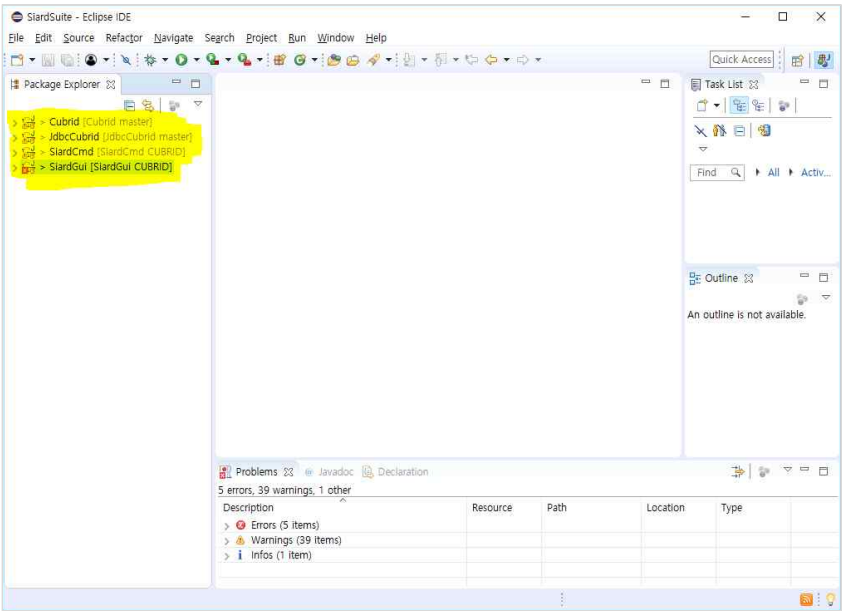
② 다운로드한 4개의 Repository를 Eclipse의 Project에 포함

③ Cubrid Repository (File 메뉴 → Open Project From File System → Directory → D:\SiardSuite\Cubrid 폴더 선택) → Finish

④ JdbcCubrid Repository (File 메뉴 → Open Project From File System → Directory → D:\SiardSuite\JdbcCubrid 폴더 선택) → Finish

⑤ SiardCmd Repository (File 메뉴 → Open Project From File System → Directory → D:\SiardSuite\SiardCmd 폴더 선택) → Finish

⑥ SiardGui Repository (File 메뉴 → Open Project From File System → Directory → D:\SiardSuite\SiardGui 폴더 선택) → Finish



<그림 45> 4개의 Repository을 Project로 포함한 Eclipse 화면

4. Siard Suite Extension for CUBRD Build

① Siard Cmd Properties 설정 (SiardCmd 오른쪽 마우스 클릭 → Properties - Java Build Path → Projects → Add → Cubrid, JdbcCubrid 선택 → OK)

② Siard Gui Properties 설정 (SiardGui 오른쪽 마우스 클릭 → Properties - Java Build Path → Projects → Add → Cubrid, JdbcCubrid, SiardCmd 선택 → OK)

③ Siard Gui Properties 설정 (SiardGui 오른쪽 마우스 클릭 → Properties - Java Build Path → Libraries → “siardcmd.jar” Remove → Apply and Close)

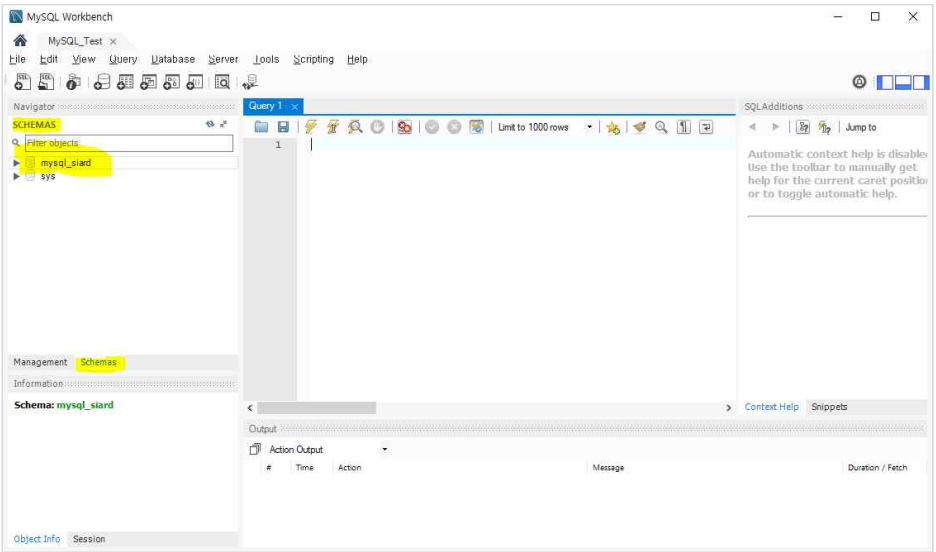
④ Siard Gui 실행(SiardGui 오른쪽 마우스 클릭 → Run As → Java Application → SiardGui - ch.admin.bar.siard2.gui 선택 → OK

<그림 46> SiardGui 실행화면

5. Siard Suite Extension for CUBRD Build Download 방법(MySQL Workbench 기준)

※ MySQL을 제외한 나머지 DBMS 사용방법은 [별첨03] SIARD Suite 빌드 및 SiardGui 실행 방법 참고

① Download할 Schema 생성(Schema 명은 mysql_siard라고 가정, create schema mysql_siard 입력)

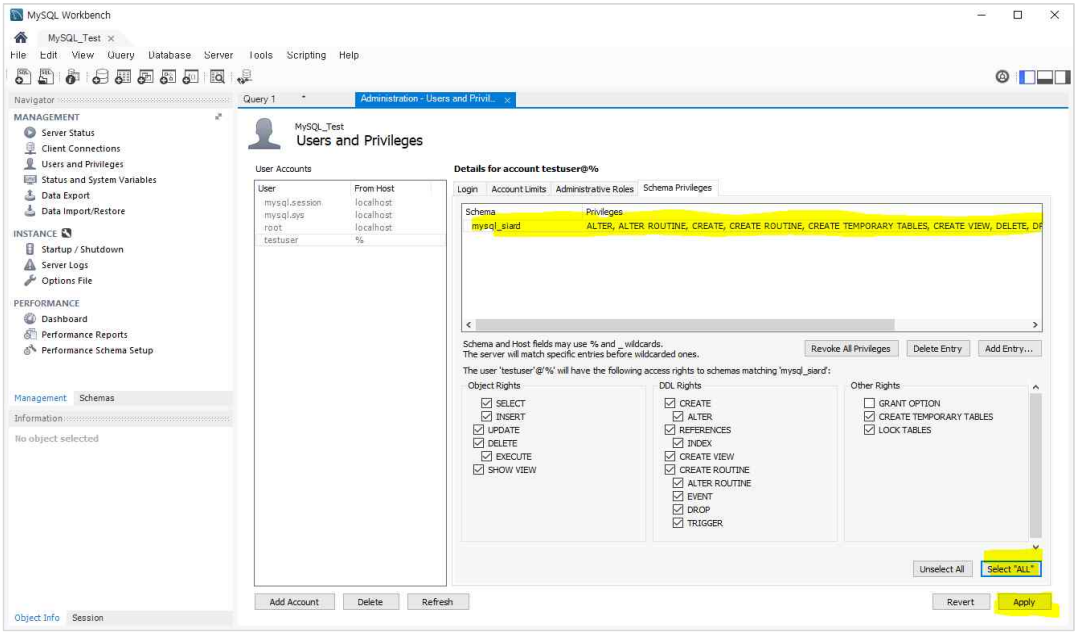


<그림 47> MySQL Workbench 첫 화면

② mysql_siard Schema에 접근 권한을 가진 user 생성

③ Management → User and Privileges → Add Account → Login Name: testuser, Password 입력 → Apply

④ Schema Privileges → Add Entry, Select Schema → mysql_siard, Select All → Apply



<그림 48> testuser에게 mysql_siard schema 접근 권한을 준 화면

⑤ SiardGui 실행 및 Download 설정값 입력(SiardGui 실행 → File 메뉴 → Download)

구분	의미	입력값
Database Management System	DBMS 종류	· MySQL
Database server	DBMS IP주소	· localhost
Database name	DB Schema 이름	· mysql_siard
JDBC URL for database connection, e.g.	JDBC URL	· jdbc:mysql://localhost:3306/mysql_siard?zeroDateTimeBehavior=convert_To_Null&characterEncoding=UTF-8&serverTimezone=Asia/Seoul
Database user	사용자 계정	· testuser
Database password	사용자 비밀번호	· 계정생성시 설정한 비밀번호 · (여기서는 “123456”)
Meta data only	Meta data만 DownLoad	· 해제(☐)
Archive views as tables	Archive Views로	· 해제(☐)

<표 75> Download를 위해 설정해야 하는 값

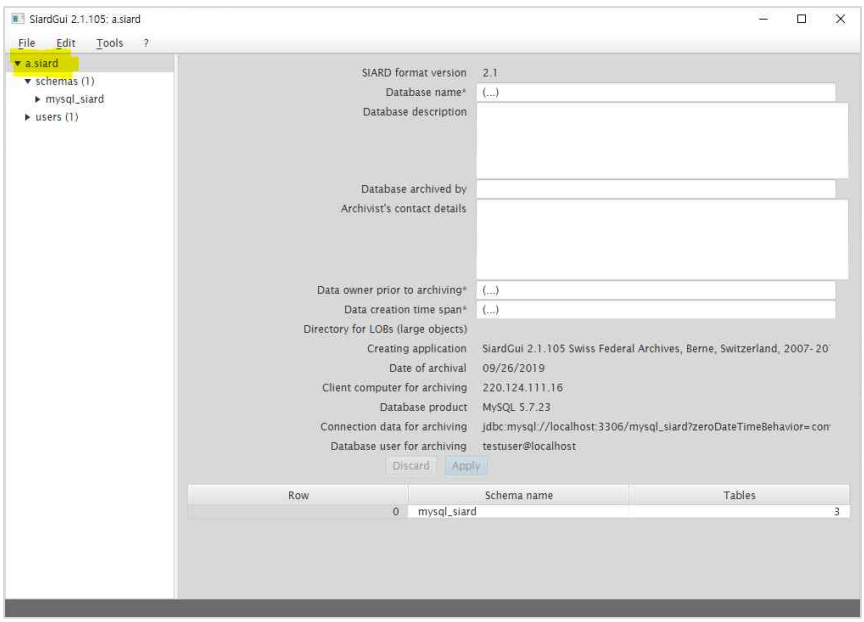
<그림 49> Download 설정값 입력 모습

⑥ 파일 이름 설정 → OK

6. Siard Suite Extension for CUBRD Build Upload 방법(MySQL Workbench 기준)

① MySQL에서 Upload 대상 schema, 이에 접근할 수 있는 user 생성

② SiardGui 실행 및 Open(SiardGui 실행 → File 메뉴 → Open → Upload할 SIARD 파일 선택 → OK)



<그림 50> Upload할 SIARD 파일을 open한 모습

③ SiardGui 실행 및 Upload(File 메뉴 → Upload → 설정값 입력)

구분	의미	입력값
Database Management System	DBMS 종류	· MySQL
Database server	DBMS IP주소	· localhost
Database name	DB Schema 이름	· mysql_siard_up
JDBC URL for database connection, e.g.	JDBC URL	· jdbc:mysql://localhost:3306/mysql_siard_up?zeroDateTimeBehavior=convert_To_Null&characterEncoding=UTF-8&serverTimezone=Asia/Seoul
Database user	사용자 계정	· testuser_up
Database password	사용자 비밀번호	· 계정생성시 설정한 비밀번호
Schema only	Schema만 Upload	· 해제(☐)
Overwrite types and tables	타입과 테이블 덮어쓰기	· 설정(☑)
Schema mapping or upload	Upload할 DB Schema 이름	· mysql_siard_up

<표 76> Upload를 위해 설정해야 하는 값

4.2 보존포맷 변환 검증 사전 시험 결과

4.2.1 보존포맷 변환 검증 사전 시험 결과 요약

○ 보존포맷 변환 검증 사전 시험 결과 요약(<표 77> 참고)

- 4종 DBMS를 대상으로 진행한 보존포맷 변환 검증 사전 시험의 결과를 요약한 표는 아래와 같음
- Data Type의 경우, 숫자. 문자, Large Object. 날짜 및 시간의 Data Type은 일반 Data Type으로 설정
- 일반 Data Type에 포함하지 않는 Data Type은 특수 Data Type으로 설정
- Table 간의 관계를 보여주는 PK, FK를 임의로 설정한 후 검증 사전 시험 진행
- Routine Type도 임의로 생성하여 검증 사전 시험을 진행하였음

항목 \ DBMS	MySQL	SQL Server	Oracle	큐브리드	
				SW개발 전	SW개발 후
일반 Data Type	◎	◎	◎	X	◎
특수 Data Type	○	◎	○	X	◎
Key Type (PK, FK)	◎	◎	○	X	◎
Routine Type	X	X	X	X	◎

<표 77> 4종 DBMS↔SIARD 사전 시험 결과 요약표

(◎: 모두 변환 가능, O: 부분 변환 가능, X: 변환 불가능)

4.2.2 MySQL↔SIARD 사전 시험 결과

○ Data Type 변환 및 PK, FK 변환 결과

- MySQL 제작사에서 배포한 매뉴얼을 참고하여 MySQL의 Data Type을 SIARD로 변환을 시도함
- 일반 Data Type은 SIARD로 정상적인 변환이 가능함
- 특수 Data Type 중 하나인 “JSON” Data Type은 SIARD로 변환이 불가능한 것을 확인
- Download 하려는 Schema에 “JSON” Data Type이 포함되는 경우, 전체 Schema가 Download 되지 않음
- MySQL의 Data Type과 SIARD(SQL:2008) 매핑 결과는 아래의 <표 78> 참고
- MySQL DB의 Key Type은 정상적으로 변환되는 것을 확인

종류	Data Type	
	MySQL	SIARD(SQL:2008)
숫자	BIT	BOOLEAN
	INT	INTEGER
	TINYINT	SMALLINT
	SMALLINT	
	MEDIUMINT	INTEGER
	BIGINT	BIGINT
	NUMERIC	DECIMAL
	DECIMAL	
	DOUBLE	DOUBLE PRECISION
	REAL	
	FLOAT	FLOAT
	BOOLEAN (SIARD에서 TINYINT로 인식)	SMALLINT
문자/ 이진	CHAR	CHARACTER
	VARCHAR	VARCHAR
	BINARY	BINARY
	VARBINARY	VARBINARY
Large Object	TINYBLOB	
	BLOB	BLOB
	MEDIUMBLOB	
	LONGBOLB	
	TINYTEXT	VARCHAR
	TEXT	CLOB
	MEDIUMTEXT	
	LONGTEXT	

Object	ENUM	VARCHAR
	SET	
날짜/시간	DATE	DATE
	TIME	TIME
	DATETIME	TIMESTAMP
	TIMESTAMP	
	YEAR	SMALLINT
특수	JSON	변환 불가
	GEOMETRY	CLOB
	POINT	
	MULTIPOINT	
	LINESTRING	
	MULTILINESTRING	
	POLYGON	
	MULTIPOLYGON	
	GEOMETRYCOLLECTION	

<표 78> MySQL↔SIARD Data Type 매핑 결과

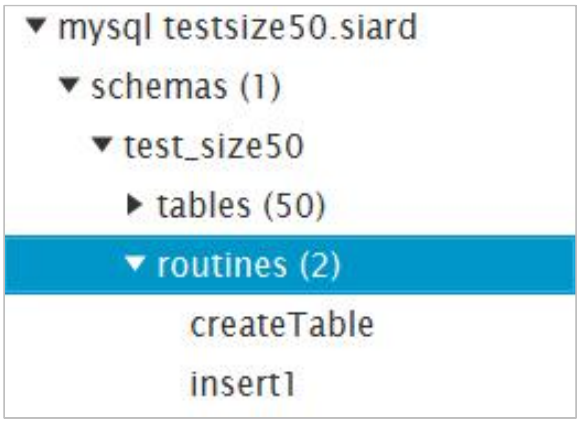
○ Routine Type 변환 결과

- MySQL의 Routine Type(Trigger, Stored Procedures / Function)을 SIARD로 변환을 할 경우, “routines” 카테고리에 Routine Type의 이름 정보만 변환되는 것을 확인 (<그림 51> 참고)

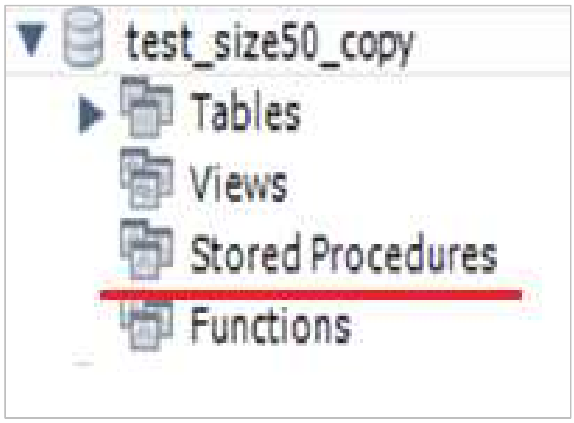
- MySQL로 업로드를 진행하면 Routine Type의 이름 정보를 포함한 모든 정보가 누락 되는 것을 확인 (<그림 52> 및 <그림 53> 참고)

```
▼<routines>
  ▼<routine>
      <specificName>createTableProcTest</specificName>
      <name>createTableProcTest</name>
      <description/>
    </routine>
  </routines>
```

<그림 51> MySQL의 Routine Type의 SIARD 변환 모습



<그림 52> MySQL의 Routine Type의 SIARD 변환



<그림 53> MySQL의 Routine Type Upload 화면

4.2.3 SQL Server↔SIARD 사전 시험 결과

○ Data Type 변환 및 Key Type 변환 결과

- SQL Server 제작사에서 배포한 매뉴얼을 참고하여 SQL Server의 Data Type을 SIARD로 변환을 시도함
- SQL Server의 일반, 특수 Data Type 모두 SIARD로 정상적인 변환이 가능함
- SQL Server의 Data Type과 SIARD(SQL:2008) 매핑 결과는 아래의 <표 79> 참고
- SQL Server DB의 PK, FK도 정상적으로 변환되는 것을 확인

종류	Data Type	
	SQL Server 2014	SIARD(SQL:2008)
숫자	bit	boolean
	int	integer
	tinyint	smallint
	smallint	
	bigint	bigint
	money	decimal
	smallmoney	
	numeric	numeric
	decimal	decimal
	float	double precision
	real	real

문자	char	character
	nchar	nchar
	varchar	varchar
	nvarchar	nchar varying
Large Object /이진	binary	binary
	varbinary(max)	varbinary
	text	clob
	ntext	nclob
	image	blob
날짜/ 시간	date	date
	time	time
	datetime	timestamp
	datetime2	
	datetimeoffset	varchar
	smalldatetimeoffset	timestamp
특수	geography	varchar
	geometry	varchar

<표 79> SQL Server↔SIARD Data Type 매핑 결과

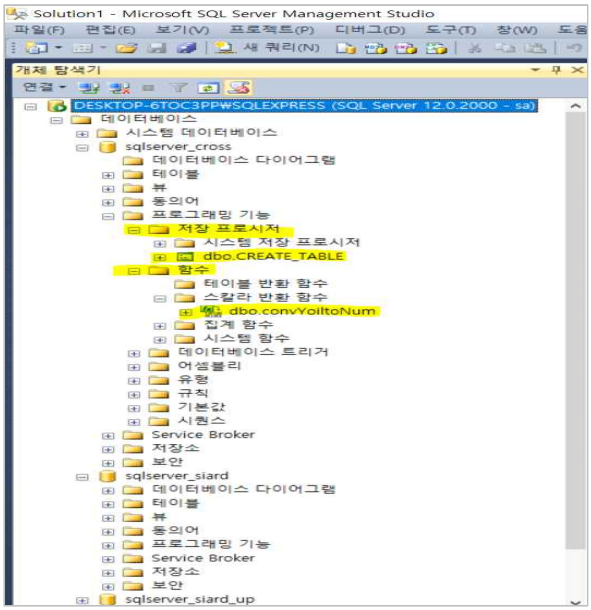
○ Routine Type 변환 결과

- SQL Server의 Routine Type(Trigger, Stored Procedures, Function)을 SIARD로 변환을 할 경우, “routines” 카테고리에 Routine Type의 이름 정보만 변환되는 것을 확인 (<그림 54> 참고)

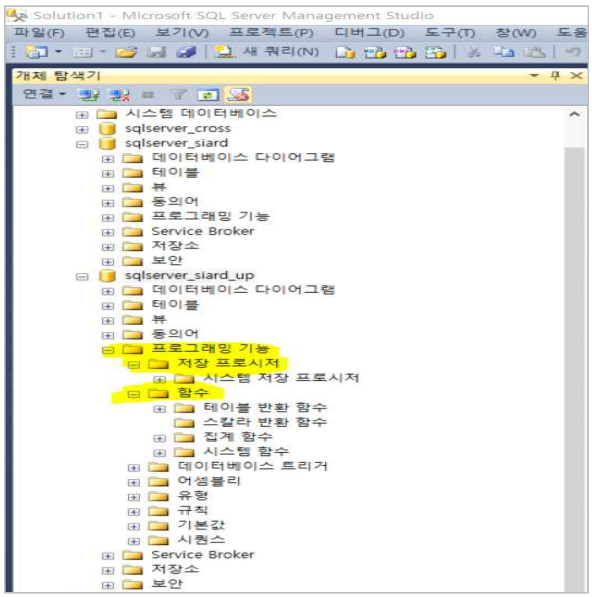
- SQL Server로 업로드를 진행하면 Routine Type의 이름 정보를 포함한 모든 정보가 누락되는 것을 확인(<그림 55> 및 <그림 56> 참고)

```
<routine>
    <specificName>CREATE_TABLE;1</specificName>
    <name>CREATE_TABLE;1</name>
    <characteristic>PROCEDURE</characteristic>
</routine>
```

<그림 54> SQL Server의 Routine Type의 SIARD 변환 모습



<그림 55> SQL Server의 Routine Type의 생성 화면



<그림 56> SQL Server의 Routine Type 업로드 화면

4.2.4 Oracle↔SIARD 사전 시험 결과

○ Data Type 변환 및 Key Type 변환 결과

- Oracle 제작사에서 배포한 매뉴얼을 참고하여 Oracle의 Data Type을 SIARD로 변환을 시도함
- Oracle의 대부분 Data Type은 SIARD로 정상적인 변환이 가능함
- “UROWID” Data Type은 SIARD로 변환이 불가능한 것을 확인
- Download 하려는 DB에 “UROWID”이 포함되어 있는 경우, 전체 DB가 Download 되지 않음
- Oracle의 Data Type과 SIARD(SQL:2008) 매핑 결과는 아래의 <표 80> 참고
- Oracle DB에 설정한 Key Type은 SIARD로 정상적으로 변환이 되었으나, Oracle로 Upload할 경우 PK는 정상적으로 변환이 되지만 FK는 누락되는 것을 확인(<그림 59 >참고)
- FK의 경우, Oracle로 Upload 시 업로드DB의 PK 제약조건 이름(Constraint Name)이 DBMS(Oracle SQL Developer) 임의로 변경되어 원본DB의 PK와 업로드DB의 PK를 동일한 것으로 인식하지 못함(<그림 57>, <그림 58> 참고)
- 따라서 변경된 제약조건 이름을 가진 PK에 영향을 받아 FK는 누락되는 것
- Routine Type은 대부분의 정보가 누락되고 이름 정보만 SIARD로 변환이 되었으나, DBMS로 Upload할 경우 이름 정보마저 누락되어 Upload 되는 것을 확인

	CONSTRAINT_NAME	CONSTRAINT_TYPE
1	FK_PUB_ID_PUBLISHED_PUB_ID	Foreign_Key
2	FK_WRI_ID_WRITERS_WRI_ID	Foreign_Key
3	SYS_C0012338	Check
4	SYS_C0012339	Primary_Key

<그림 57> 원본 DB의 BOOK table 제약조건(PK, FK)

	CONSTRAINT_NAME	CONSTRAINT_TYPE
1	SYS_C0012346	Check
2	SYS_C0012347	Primary_Key

<그림 58> 업로드 DB의 BOOK table 제약조건(PK)

```
<primaryKey>
    <name>SYS_C0012339</name>
    <column>BOOK_ID</column>
</primaryKey>
<foreignKeys>
    <foreignKey>
        <name>FK_PUB_ID_PUBLISHED_PUB_ID</name>
        <referencedSchema>USER1</referencedSchema>
        <referencedTable>PUBLISHED</referencedTable>
        <reference>
```

<그림 59> 원본DB SIARD파일 부분 발췌

종류	Data Type	
	Oracle 11g	SIARD 2.1(SQL:2008)
숫자	NUMBER	DECIMAL
	FLOAT	FLOAT
	BINARY_FLOAT	REAL
	BINARY_DOUBLE	DOUBLE PRECISION
문자	CHAR	CHAR
	VARCHAR2	VARCHAR
	NCHAR	NCHAR
	NVARCHAR2	NCHAR VARYING
Large Object	LONG	CLOB
	RAW	VARBINARY
	LONG RAW	BLOB
	BLOB	
	BFILE	
	CLOB	CLOB
	NCLOB	NCLOB
날짜/시간	DATE	DATE
	TIMESTAMP	TIMESTAMP
	TIMESTAMP WITH TIME ZONE	
	TIMESTAMP WITH LOCAL TIME ZONE	
	INTERVAL YEAR TO MONTH	INTERVAL YEAR TO MONTH
	INTERVAL DAY TO SECOND	INTERVAL DAY TO SECOND
특수	ROWID	BIGINT
	UROWID	변환불가

<표 80> Oracle↔SIARD Data Type 매핑 결과

○ Routine Type 변환 결과

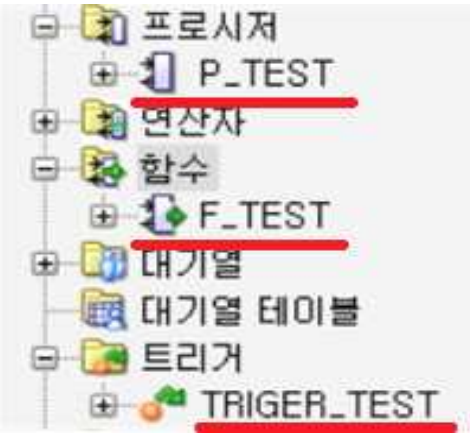
- Oracle의 Routine Type(Trigger, Stored Procedures, Function)을 SIARD로 변환을 할 경우, “routines” 카테고리에 Routine Type의 이름 정보만 변환되는 것을 확인 (<그림 60>, <그림 61> 참고)

- Oracle로 업로드를 진행하면 Routine Type의 이름 정보를 포함한 모든 정보가 누락 되는 것을 확인(<그림 62> 참고)

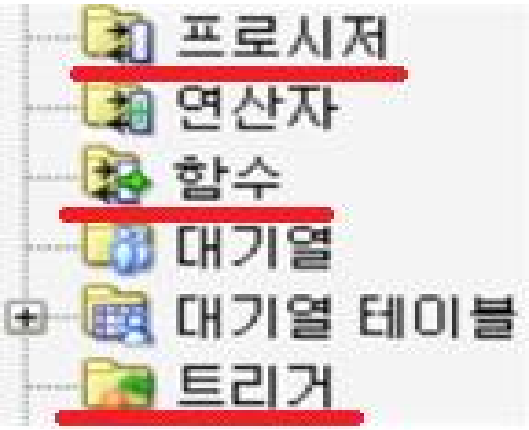
-

```
<routines>
  <routine>
    <specificName>F_TEST</specificName>
    <name>F_TEST</name>
    <returnType>VARCHAR</returnType>
  </routine>
  <routine>
    <specificName>P_TEST</specificName>
    <name>P_TEST</name>
  </routine>
  <routine>
    <specificName>TRIGER_TEST</specificName>
    <name>TRIGER_TEST</name>
  </routine>
</routines>
```

<그림 60> Oracle의 Routine Type SIARD 변환 모습



<그림 61> Oracle의 Routine Type 변환 전



<그림 62> Oracle의 Routine Type Upload 화면

4.2.5 큐브리드↔SIARD 사전 시험 결과

○ 큐브리드 확장 버전 개발 전 상황

- 기존의 SIARD는 국산 DBMS인 큐브리드를 지원하지 않음
- 그렇기 때문에, SIARD의 기능이 큐브리드를 지원할 수 있도록 지원 범위를 확장하는 개발을 한 뒤에 사전 시험 진행

○ 큐브리드 확장 버전 개발 후, Data Type 변환 및 PK, FK 변환 결과

- 큐브리드 제작사에서 배포한 매뉴얼을 참고하여 큐브리드의 Data Type을 SIARD로 변환을 시도함
- 큐브리드의 일반, 특수 Data Type 모두 SIARD로 정상적인 변환이 가능함
- 큐브리드의 Data Type과 SIARD(SQL:2008) 매핑 결과는 아래의 <표 81> 참고
- 큐브리드 DB에 설정한 PK, FK도 정상적으로 변환이 되는 것을 확인

○ 큐브리드 확장 버전 개발 후, Routine Type 변환 결과

- 큐브리드의 Routine Type은 정상적으로 변환이 되는 것을 확인
- 큐브리드의 Routine Type은 JAVA class 파일로 이뤄져 있으며, 본 연구 과제에서는 이러한 큐브리드 DB의 구조적 특성에 맞춰서 큐브리드 확장 버전을 개발하였음
- 이와 관련된 내용은 제5장에서 상세히 다룰 예정

종류	Data Type	
	큐브리드	SIARD(SQL:2008)
숫자	SHORT, SMALLINT	SMALLINT
	INTEGER	INTEGER
	BIGINT	BIGINT
	NUMERIC	NUMERIC
	FLOAT	FLOAT
	DOUBLE	DOUBLE
날짜/시간	DATE	DATE
	TIME	TIME
	TIMESTAMP	TIMESTAMP
	DATETIME	TIMESTAMP

비트열	BIT	VARCHAR
	BIT VARYING	VARCHAR
문자	CHAR	CHAR
	VARCHAR	VARCHAR
	STRING	VARCHAR
Object	ENUM	VARCHAR
Large Object	BLOB	BLOB
	CLOB	CLOB
Collection (특수)	SET	VARCHAR
	MULTISET	VARCHAR
	LIST, SEQUENCE	VARCHAR

<표 81> 큐브리드↔SIARD Data Type 매핑 결과

4.2.6 보존포맷 변환 검증 사전 시험 한계점

○ 보존포맷 변환 검증 사전 시험의 한계점

- 보존포맷 변환 검증 사전 시험은 자체 검증 대상 DBMS인 MySQL, SQL Server, Oracle, 큐브리드의 일반, 특수 Data Type과 PK, FK, Routine Type 등 여러 항목을 SIARD 변환 여부를 중점적으로 확인함

- DBMS에서 SIARD로의 변환 여부를 중점적으로 확인하기 위해 사전 시험을 진행하였기 때문에 각각의 DBMS별로 공통적이지 않은 DB를 생성해 진행

- DBMS별로 서로 다른 DB를 구성하였기 때문에 사전 시험에 일관성이 없다고 판단

- 또한, DBMS에서 SIARD로 변환 여부에 초점을 맞췄기에, DB에서 가장 중요한 항목인 Data가 변환 후에도 변화가 발생하지 않는지 검증하는 과정을 생략하였음

- 따라서 SIARD 변환 후 Data의 변화 여부를 확인 및 검증하는 추가적인 시험이 필요하다고 판단됨

- 이에 보존포맷 변환 검증 시험에서는 DBMS 유형에 따라, TOAD Data Point와 CUBRID MANAGER의 비교 마법사 TOOL을 이용해 변환 전의 DB(원본DB)와 SIARD 변환 과정을 거쳐 다시 DB 형태를 가진 DB(업로드DB)의 Data의 값을 비교하는 시험을 진행

4.3 보존포맷 변환·복원 검증 시험 개요

- 검증 시험은 사전 시험의 한계점에서 나타난 Data 동일 여부를 확인하는 것을 목적으로 진행
- 4종 DBMS에 최대한 유사한 공통 DB를 기준으로 하여, SIARD 포맷으로 변환을 한 뒤 Data 변화 여부 확인
- SIARD Suite을 이용해 정상적인 변환이 가능한 DB Size를 파악하기 위해 1~7GB의 DB에 대해 변환 검증 수행
- Data 동일 여부를 검증하기 위한 Tool은 DBMS 종류에 따라 TOAD Data Ponint, CUBRID MANAGER 비교 마법사 TOOL을 이용함

4.3.1 보존포맷 변환·복원 검증 시험 목적

○ 보존포맷 변환·복원 검증 시험 목적

- 보존포맷 변환·복원 검증 시험은 검증 대상 DBMS에 공통 DB를 각각 생성하여, SIARD 포맷으로 변환을 한 뒤 Data 변환 여부를 확인하고자 함
- DB Size별 Download 및 Upload 소요시간을 측정하여, 추후 사용자가 SiardSuite을 사용하여 SIARD 포맷으로 변환하고자 할 때 소요시간을 예측할 수 있도록 기초자료를 제공하고자 함

4.3.2 보존포맷 변환·복원 검증 시험 환경

○ 보존포맷 변환·복원 검증 시험 환경 구축

- 효율적인 보존포맷 변환·복원 검증 시험과 더불어 실데이터 검증을 대비하여 노트북을 이용해 시험 환경 구축
- 구축한 보존포맷 변환·복원 검증 시험 환경의 정보는 아래의 <표 82>과 같음

제조사	CPU	RAM	SSD	HDD
HP	i7-8750H 2.2GHz	32GB	1TB	1TB

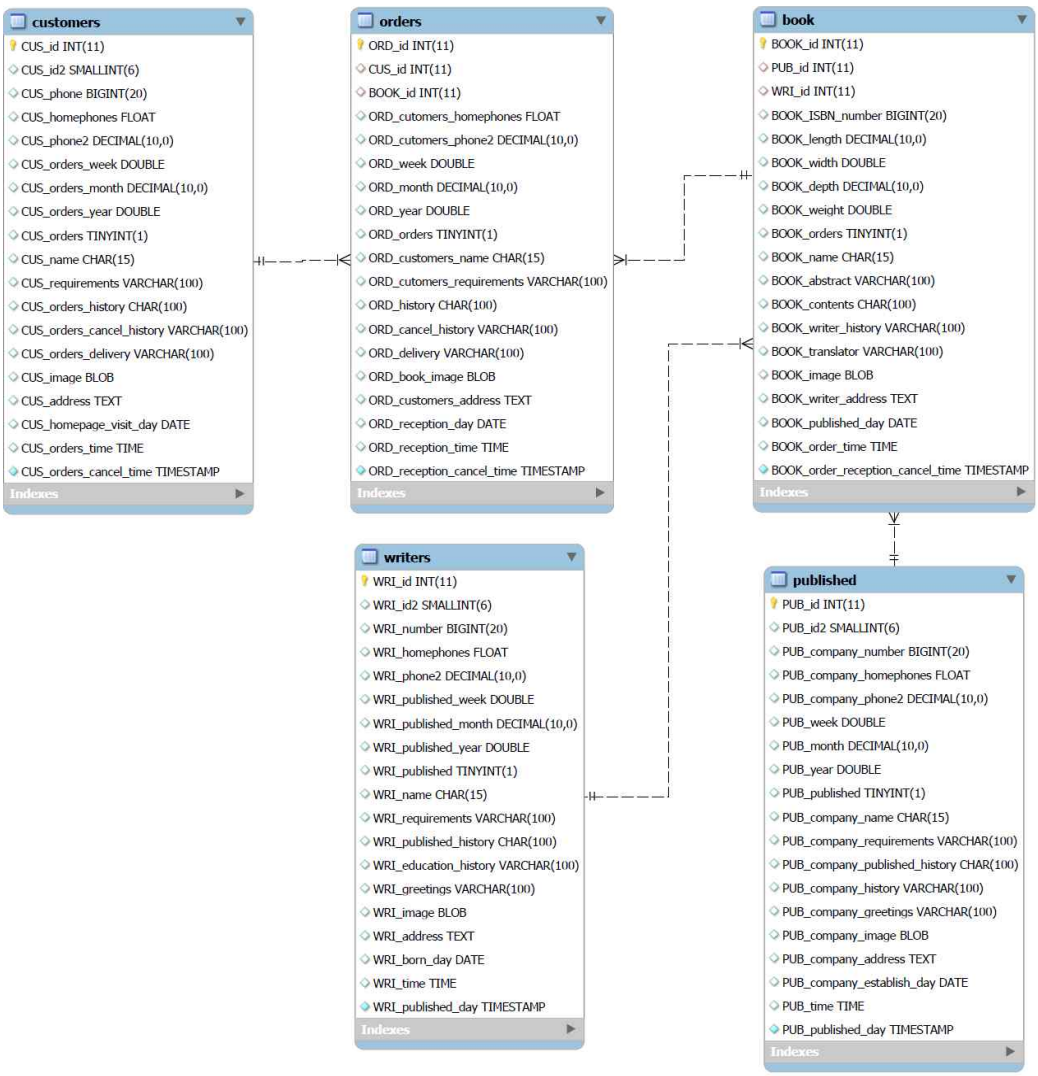
<표 82> 보존포맷 변환·복원 검증 시험 환경 정보

○ 공통 DB 소개

- 보존포맷 변환·복원 검증 시험은 자체적으로 DB를 생성하여 진행함
- 아래 <표 83>은 생성한 공통 DB에 대한 규모 정보를 보여줌
- 아래 <그림 63>은 각각의 DBMS별로 제작한 공통 DB에 대한 ERD
- 보존포맷 변환·복원 검증 시험 대상인 4종의 DBMS(MySQL, Oracle, SQL Server, 큐브리드)별로 Data Type이 다르기 때문에, 최대한 유사하며 많은 종류의 Data Type을 이용하여 각각 DBMS 마다 공통 DB를 제작함(<표 84> 참고)

테이블 수	컬럼 수	레코드 수
5개	19개	테이블별 100개

<표 83> 공통 DB 규모 정보



<그림 63> 공통 DB의 ERD

Data Type					
	SQL:2008	MySQL	Oracle	SQL Server	큐브리드
1	INT	INT	NUMBER	INT	INT
2	SMALLINT	SMALLINT	NUMBER	SMALLINT	SMALLINT
3	BIGINT	BIGINT	NUMBER	BIGINT	BIGINT
4	FLOAT	FLOAT	FLOAT	FLOAT	FLOAT
5	DECIMAL	DECIMAL	NUMBER	DECIMAL	NUMERIC
6	DOUBLE	DOUBLE	BINARY DOUBLE	DOUBLE	DOUBLE
7	DECIMAL	DECIMAL	NUMBER	NUMERIC	NUMERIC
8	DOUBLE	DOUBLE	BINARY DOUBLE	REAL	DOUBLE
9	SMALLINT	TINYINT	NUMBER	SMALLINT	SMALLINT
10	CHAR	CHAR	CHAR	CHAR	CHAR
11	VARCHAR	VARCHAR	VARCHAR2	VARCHAR	VARCHAR
12	NCHAR	CHAR	NCHAR	NCHAR	CHAR
13	VARCHAR	VARCHAR	VARCHAR2	VARCHAR	VARCHAR
14	VARCHAR	VARCHAR	VARCHAR2	VARCHAR	VARCHAR
15	BLOB	BLOB	BLOB	VARBINARY	BLOB
16	CLOB	TEXT	CLOB	TEXT	CLOB
17	DATE	DATE	DATE	DATE	DATE
18	TIME	TIME	VARCHAR2	TIME	TIME
19	TIMESTAMP	TIMESTAMP	TIMESTAMP	DATETIME2	TIMESTAMP

<표 84> DBMS별 공통 DB Data Type

4.3.3 보존포맷 변환·복원 검증 시험 방법

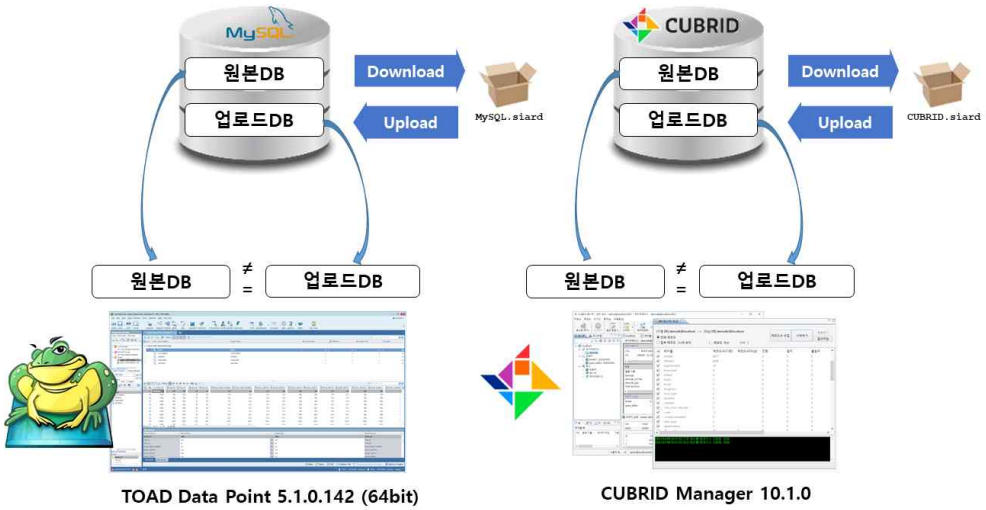
○ 보존포맷 변환·복원 검증 시험 방법(같은 DBMS일 경우)

- 같은 DBMS를 대상으로 보존포맷 변환·복원 검증 시험을 진행할 경우, 3단계로 진행(<표 85>, <그림 64> 참고)

- 데이터 비교 검증을 위해 외산 DBMS(MySQL, Oracle, SQL Server)는 TOAD Data Point를 사용하였으며, 국산 DBMS(큐브리드)는 CUBRID MANAGER의 비교 마법사 TOOL을 사용

순서	상세 내역	
1. DB 생성	· 동일한 쿼리문을 이용해 4종의 DBMS에서 각각 DB 생성	
2. SIARD 파일 생성	· 생성한 DB를 siard 파일로 변환	
3. 동일한 DBMS로 Upload 및 데이터 비교 검증	<원본DB> MySQL Oracle SQL Server	· SIARD 파일을 비교 검증하기 위해 원본 DB와 동일한 DBMS로 Upload 진행 · **TOAD Data Point** 5.1.0.142를 이용해 데이터 비교 검증 진행
	<원본DB> 큐브리드	· SIARD 파일을 비교 검증하기 위해 원본 DB와 동일한 DBMS로 Upload 진행 · **CUBRID MANAGER**에서 제공하는 **비교 마법사 TOOL**를 이용해 데이터 비교 검증 진행

<표 85> 보존포맷 변환 검증 시험 방법(같은 DBMS)



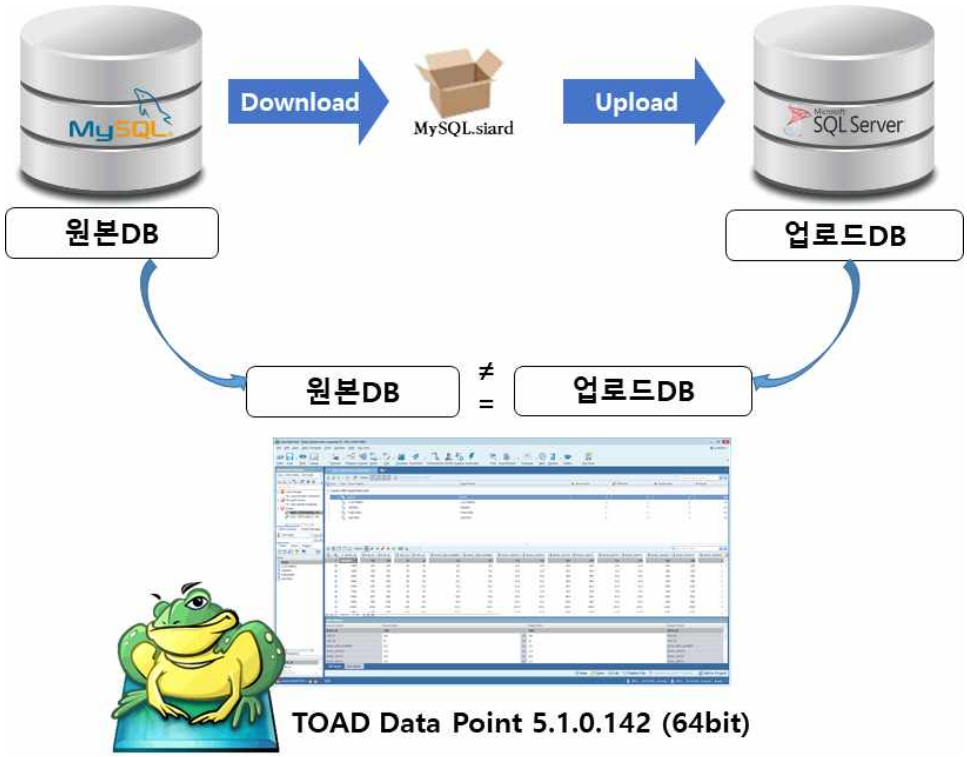
<그림 64> 보존포맷 변환·복원 검증 시험 방법(같은 DBMS)

○ 보존포맷 변환·복원 검증 시험 방법(다른 DBMS일 경우)

- 서로 다른 DBMS 대상으로 보존포맷 변환·복원 검증 시험을 진행할 경우, 4단계로 진행
- 본 변환·복원 검증 시험에서는 MySQL을 원본DB, SQL Server를 업로드DB로 선정

순서	상세 내역	
1. DB 생성	· 동일한 쿼리문을 이용해 Oracle의 원본DB 생성	
2. SIARD 파일 생성	· 생성한 DB를 SIARD 파일로 변환	
3. 다른 DBMS로 Upload 및 데이터 비교 검증	<업로드DB> SQL Server	· 원본DB(MySQL)의 SIARD 파일을 비교 검증하기 위해 업로드DB(SQL Server)로 Upload 진행 · TOAD Data Point를 이용해 데이터 비교 검증 진행

<표 86> 보존포맷 변환·복원 검증 시험 방법(다른 DBMS)



<그림 65> 보존포맷 변환·복원 검증 시험 방법(다른 DBMS)

4.3.4 보존포맷 변환·복원 검증 시험 시나리오: 외산 3종 DBMS↔SIARD

○ 외산 3종 DBMS ↔ SIARD 보존포맷 변환·복원 검증 시험

- 외산 3종 DBMS(MySQL, Oracle, SQL Server)는 동일한 순서와 방법으로 진행
- 같은 DBMS(MySQL)에서 Download, Upload를 진행한 후, TOAD Data Point를 이용하여 데이터 비교 검증을 진행
- 앞서 소개한 보존포맷 변환·복원 검증 시험 방법(같은 DBMS일 경우)과 동일하게 진행

4.3.5 보존포맷 변환·복원 검증 시험 시나리오: 큐브리드↔SIARD

○ 큐브리드 ↔ SIARD 보존포맷 변환·복원 검증 시험

- 같은 DBMS(큐브리드)에서 Download, Upload를 진행
- 이후 CUBRID MANAGER 비교 마법사 TOOL을 이용하여 원본DB와 업로드DB의 데이터 비교 검증을 진행
- 앞서 소개한 보존포맷 변환·복원 검증 시험 방법(같은 DBMS일 경우)과 동일하게 진행

4.3.6 보존포맷 변환·복원 검증 시험 시나리오 MySQL↔SQL Server

○ MySQL ↔ SQL Server 보존포맷 변환·복원 검증 시험

- 원본DB인 MySQL에서 생성한 DB를 SIARD 파일로 변환
- 이후 SQL Server로 Upload 진행
- SQL Server에 Upload한 DB와 원본DB를 TOAD Data Point를 이용하여 데이터 비교 검증을 진행
- 앞서 소개한 보존포맷 변환·복원 검증 시험 방법(다른 DBMS일 경우)과 동일하게 진행

4.3.7 보존포맷 변환·복원 검증 시험 결과

가. 4종 DBMS↔SIARD 변환·복원 검증 시험 결과 요약표

○ 4종 DBMS↔SIARD 변환·복원 검증 시험 결과 요약

- 내부 검증 대상인 4종의 DBMS를 대상으로 진행한 변환·복원 검증 시험에 사용한 Data Type은 아래 <표 87>와 같음
- SQL:2008의 Data Type을 기준으로 각각 DBMS에 유사한 Data Type을 선정하여 Data 일치 여부를 확인
- 모든 DBMS의 Data가 변환·복원 후에도 일치하는 것을 확인(<표 88> 참고)

SIARD(SQL:2008) Data Type				
정수 타입	실수 타입	문자 타입	Large Object 타입	날짜, 시간 타입
INT SMALLINT BIGINT DECIMAL	FLOAT DOUBLE	CHAR VARCHAR NCHAR	BLOB CLOB	DATE TIME TIMESTAMP

<표 87> SIARD(SQL:2008) Data Type

DBMS / 항목	MySQL	SQL Server	Oracle	큐브리드
Data	◎	◎	◎	◎

<표 88> 4종 DBMS↔SIARD 검증 결과 요약표

나. MySQL↔SIARD 변환·복원 검증 시험 결과

○ MySQL↔SIARD 변환·복원 검증 시험 결과 내용

- TOAD Data Point를 이용해 데이터 비교 검증 결과, 아래의 <그림 66>와 같이 모든 데이터가 일치한 것을 확인

Sync	Type	Source Name	Target Name	Source Only	Different	Target Only	Equal
5 pairs with equal data only							
		BOOK	BOOK	0	0	0	100
		CUSTOMERS	CUSTOMERS	0	0	0	100
		ORDERS	ORDERS	0	0	0	100
		PUBLISHED	PUBLISHED	0	0	0	100
		WRITERS	WRITERS	0	0	0	100

<그림 66> MySQL↔SIARD 데이터 비교 결과

다. SQL Server↔SIARD 변환·복원 검증 시험 결과

○ SQL Server↔SIARD 변환·복원 검증 시험 결과 내용

- TOAD Data Point를 이용해 데이터 비교 검증 결과, 아래의 <그림 67>와 같이 모든 데이터가 일치한 것을 확인

Sync	Type	Source Name	Target Name	Source Only	Different	Target Only	Equal
5 pairs with equal data only							
		dbo.BOOK	dbo.BOOK	0	0	0	100
		dbo.CUSTOMERS	dbo.CUSTOMERS	0	0	0	100
		dbo.ORDERS	dbo.ORDERS	0	0	0	100
		dbo.PUBLISHED	dbo.PUBLISHED	0	0	0	100
		dbo.WRITERS	dbo.WRITERS	0	0	0	100

<그림 67> SQL Server↔SIARD 데이터 비교 결과

라. Oracle↔SIARD 변환·복원 검증 시험 결과

○ Oracle↔SIARD 변환·복원 검증 시험 결과 내용

- TOAD Data Point를 이용해 데이터 비교 검증 결과, 아래의 <그림 68>와 같이 모든 데이터가 일치한 것을 확인

Sync	Type	Source Name	Target Name	Source Only	Different	Target Only	Equal
5 pairs with equal data only							
		book	book	0	0	0	100
		customers	customers	0	0	0	100
		orders	orders	0	0	0	100
		published	published	0	0	0	100
		writers	writers	0	0	0	100

<그림 68> Oracle↔SIARD 데이터 비교 결과

마. 큐브리드↔SIARD 변환·복원 검증 시험 결과

○ 큐브리드↔SIARD 변환·복원 검증 시험 결과 내용

- CUBRID MANAGER 비교 마법사 TOOL을 이용해 데이터 비교 검증 결과, 아래의 <그림 69>과 같이 모든 데이터가 일치한 것을 확인

☑	테이블	레코드수[기준]	레코드수[대상]	진행	일치	불일치	누락
☑	book	100	100	100	100	0	0
☑	customers	100	100	100	100	0	0
☑	db_serial	0	0	0	0	0	0
☑	orders	100	100	100	100	0	0
☑	published	100	100	100	100	0	0
☑	writers	100	100	100	100	0	0

<그림 69> 큐브리드↔SIARD 데이터 비교 결과

바. MySQL↔SQL Server 변환·복원 검증 시험 결과

○ MySQL↔SQL Server 변환·복원 검증 시험 결과 내용

- 원본DB와 업로드DB의 Data Type 차이가 데이터에 미치는 영향을 파악하기 위해 진행
- TOAD Data Point를 이용해 데이터 비교 검증 결과, 아래의 <그림 70>와 같이 모든 데이터가 일치하지 않는 것으로 나타남
- “BOOK_name”, “BOOK_contents” 컬럼의 데이터가 일치하지 않는 것(보라색)으로 나타나지만 실제로는 <그림 71>과 같이 데이터가 일치하는 것을 확인(“BOOK_abstract” 컬럼은 동일한 데이터로 회색으로 표기됨)
- 따라서 모든 데이터가 일치하는 것으로 판단

Sync	Type	Source Name	Target Name	Source Only	Different	Target Only	Equal
5 pairs with differences in their data							
✓		book	dbo.book	0	100 ✓	0	0
✓		customers	dbo.customers	0	100 ✓	0	0
✓		orders	dbo.orders	0	100 ✓	0	0
✓		published	dbo.published	0	100 ✓	0	0
✓		writers	dbo.writers	0	100 ✓	0	0

<그림70> MySQL↔SQL Server 데이터 비교 결과 1

BOOK_name	BOOK_name	BOOK_abstract	BOOK_abstract	BOOK_contents	BOOK_contents
Politeria1	Politeria1	ABSTRACT1	ABSTRACT1	BOOK CONTENTS1	BOOK CONTENTS1 ...
Politeria2	Politeria2	ABSTRACT2	ABSTRACT2	BOOK CONTENTS2	BOOK CONTENTS2 ...

<그림 71> MySQL↔SQL Server 데이터 비교 결과 2

4.3.8 보존포맷 변환 검증 시험 : DB Size

○ 보존포맷 변환 검증 시험 DB Size 개요

- SIARD Suite을 이용해 정상적인 변환이 가능한 DB Size를 파악하고자 함
- DB Size의 경우, DBMS의 종류와 관계가 없다고 판단하여 SQL Server를 대상으로 진행
- SIARD Suite의 구조상 한번 시작한 변환 작업은 일시 정지가 되지 않으며, 중간에 문제가 발생할 경우 처음부터 변환 작업을 시작해야 하는 등 안정성에 관련한 이슈가 존재함
- 따라서 SIARD Suite의 안정적인 변환작업을 위해 1일 근무시간(8시간)을 기준으로 하여 1일 근무시간 안에 Download, Upload가 가능한 규모를 파악함
- DB Size 검증 시험에 사용한 DB 정보와 시험 결과는 아래 <표 89>과 같음
- 시험 결과에 따라 최소 10GB, 301만 개 규모의 DB는 안정적으로 SIARD 포맷 변환이 가능하다고 판단됨
- Download 소요시간에 비해, Upload 소요시간이 상대적으로 많이 짧은 것으로 나타났음

DB Size	table 개수	총 row 개수	Download 소요시간	Upload 소요시간
약 1GB (1,112MB)	5개	총 331,500개	약 20분	약 2분
약 2GB (2.205MB)	5개	총 662,500개	약 40분	약 4분
약 3GB (3,416MB)	5개	총 823,500개	약 1시간	약 6분
약 4GB (4,262MB)	5개	총 994,500개	약 1시간 15분	약 9분
약 5GB (5,300MB)	5개	총 1,123,500개	약 1시간 40분	약 13분
약 6GB (6,424MB)	5개	총 1,476,500개	약 2시간	약 16분
약 7GB (7,451MB)	5개	총 1,861,500개	약 2시간 30분	약 20분
약 8GB (9,046MB)	5개	총 2,264,500개	약 2시간 50분	약 28분
약 9GB (10,125MB)	5개	총 2,682,500개	약 3시간 15분	약 35분
약 10GB (11,864MB)	5개	총 3,014,500개	약 3시간 35분	약 40분

<표 89> “DB Size 시험” : DB Size별 Download, Upload 소요시간 요약표

○ SIARD 파일의 최대 크기

- SIARD는 다수의 텍스트(xsd, xml) 파일과 다수의 바이너리 파일들을 하나의 zip 파일로 압축하여 저장하는 구조로 되어 있음
- 현재 SIARD2.1 표준에도 SIARD 파일의 최대 크기는 별도로 제한하지 있지 않으며, 텍스트 파일 포맷 자체도 최대 크기는 별도로 제한되어 있지 않음
- 그러므로, SIARD 파일의 최대 크기에 영향을 미치는 것은 zip 파일이 허용하는 파일의 최대 크기와 OS의 파일 시스템에서 허용하고 있는 파일의 최대 크기임
- zip 파일의 최대 크기(<표 90> 참고)와 OS의 파일 시스템의 최대 크기(<표 91> 참고) 중 작은 값으로 판단됨

zip 유형	zip	zip64
최대 파일 크기	4GB	16 EB ((2^64 - 1) Byte)

<표 90> zip과 zip64 의 최대 파일 크기

파일 시스템	FAT16	FAT32	exFAT	Ext2	Ext3	Ext4	NTFS	XFS	GFS2	HFS	HFS+	ZFS
크기	2GB	4GB	127PB	2TB	2TB	16TB	16EB	8EB	8EB	2GB	8EB	16EB

<표 91> 파일 시스템들의 지원하는 최대 파일 크기

4.3.9 보존포맷 변환·복원 관련 이슈 및 개선방안

가. 보존포맷 변환·복원 관련 이슈

○ SIARD 포맷으로 변환이 불가능한 항목

- 보존포맷 변환 검증 시험을 통해 4종의 DBMS 중 SIARD 포맷으로 변환이 불가능한 항목들을 도출하였음
- 특수 Data Type 중 MySQL의 “JSON”, Oracle의 “UROWID”는 SIARD로 변환 불가능
- Oracle의 외래 키(FK)는 SIARD 포맷으로 변환은 가능하지만, DBMS의 구조적인 문제로 인하여 Upload 시에는 누락이 되는 것을 확인
- 큐브리드를 제외한 MySQL, SQL Server, Oracle의 Routine Type은 SIARD로 변환 시 이름 정보를 제외한 모든 정보를 누락함(Upload 시 이름 정보마저 누락)

○ SIARD 포맷의 안정성

- SIARD 포맷으로 Download 또는 Upload를 진행할 때 네트워크 또는 하드웨어적인 오류가 발생하면 처음부터 다시 Download, Upload를 진행해야 함
- 또한, Download 및 Upload와 같은 변환 과정 중 일시 정지를 할 수 없음
- 대규모 DB를 SIARD 포맷으로 변환할 때, 상당한 시간이 소요되므로 변환 과정 중 문제가 발생하면 다시 변환 작업을 진행해야 하는 안정성 관련한 문제가 존재함

나. 개선방안

○ 이슈와 관련한 개선방안

- 현재의 SIARD 버전에서는 변환이 불가능한 항목들을 정상적으로 변환을 할 수 있도록 추가적인 SW 개발
- SIARD 포맷의 안정성을 확보하기 위해 Download 및 Upload 시 일시 정지 기능 추가

5. 실데이터에 대한 보존포맷 변환·복원 검증

- 본 연구에서는 2개 종류의 실데이터(정부청사관리본부 큐브리드 DB와 국가기록원에서 제공한 오라클 DB)를 대상으로 보존포맷 변환 · 복원 검증 수행
- 약 1GB의 큐브리드 DB는 모든 데이터가 잘 변환되었으며, CUBRID Manager에서 제공하는 데이터 비교 도구를 활용하여 비교 분석을 진행함
- 약 1.5GB의 오라클 DB는 TIMESTAMP 타입(시분초 데이터는 업로드DB에 포함안됨)을 제외한 모든 데이터가 잘 변환 · 복원되었으며, TOAD Data Point 도구를 활용하여 비교 분석을 진행함
- TIMESTAMP는 추가적인 SW개발을 통해 모든 데이터가 잘 변환 · 복원되었음을 확인함

5.1 큐브리드 DB 대상 실데이터 검증

○ 정부청사관리본부 홈페이지 DB (버전: 큐브리드 8.4)

- 총 테이블 수 / 총 레코드 수 / 데이터 용량: 199개 / 약 450만 개 / 약 1GB
- 데이터 비교: CM (CUBRID Manager)의 데이터 비교 툴을 사용

○ 결과

- Download (DBMS → SIARD 파일): 초당 평균 1.5M (로컬기준) 약 10분 정도 소요
- <그림 74>과 <그림 75>은 SiardCmd-ToDB.jar를 실행 화면임
- Upload (SIARD파일 → DBMS): 초당 평균 120K (로컬기준) 약 100분 정도 소요
- <그림 72>과 <그림 73>은 SiardGui를 실행 화면임

구분 \ 항목	Download	Upload	스키마비교	데이터비교
일반 Data Type	◎	◎	동일	동일
Key Type(PK, FK)	◎	◎	동일	동일

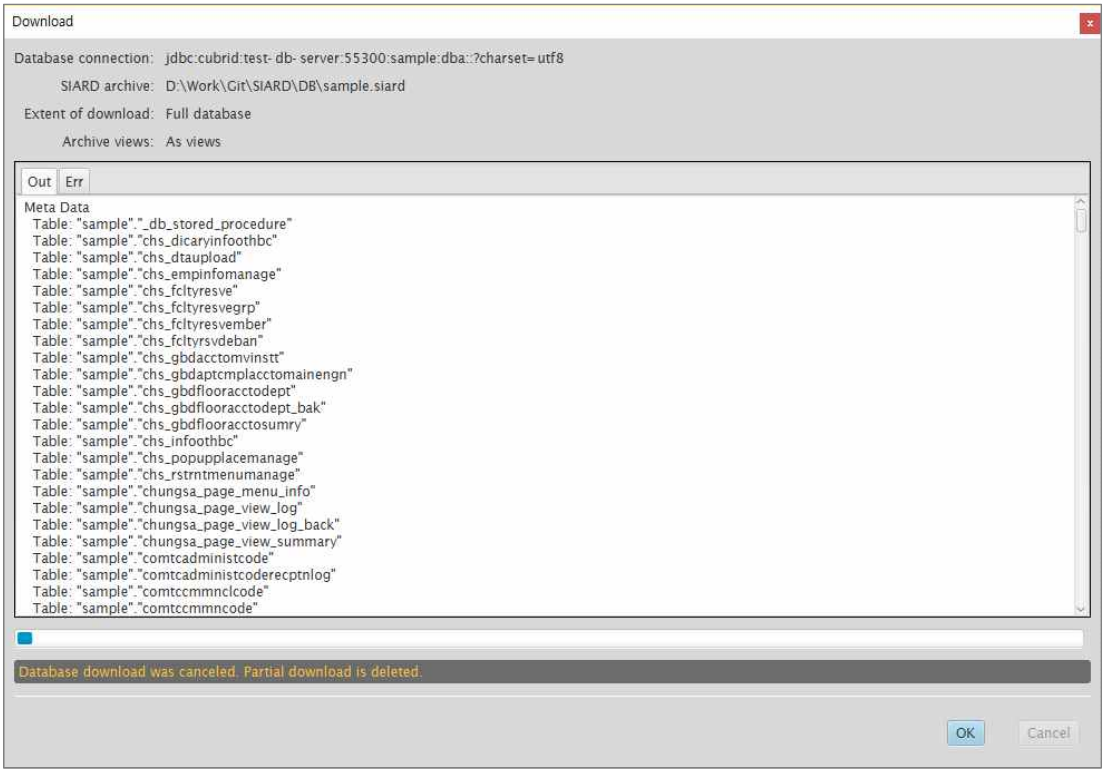
<표 92> 큐브리드 확장 SIARD Suite으로 실데이터 검증 결과>

```
java -jar SiardCmd-ToDb.jar -s=sample5.siard -j=jdbc:cubrid:localhost:55300:chungsa:dba::?charset=utf8 -u=dba -p
SiardToDb null - Program to load database content from a .siard file
SIARD Suite null: (c) Swiss Federal Archives, Berne, Switzerland, 2008-2016
Specified by : Hartwig Thomas, Enter AG, R?ti ZH, Switzerland
               Andreas Voss, Swiss Federal Archives, Berne, Switzerland
               Anders Bo Nielsen, Danish National Archives, Denmark
               Claire R?thlisberger-Jourdan, KOST, Berne, Switzerland
Developed by : Hartwig Thomas, Enter AG, R?ti ZH, Switzerland
               Simon Jutz, Cytex GmbH, Zurich, Switzerland
Tested by    : Claudia Matthys, POOL Computer AG, Zurich, Switzerland
               Marcel B?chler, Swiss Federal Archives, Berne, Switzerland
               Yvan Dutoit, Swiss Federal Archives, Berne, Switzerland
Managed by   : Hartwig Thomas, Enter AG, R?ti ZH, Switzerland
               Marcel B?chler, Swiss Federal Archives, Berne, Switzerland
               Alain Mast, Swiss Federal Archives, Berne, Switzerland
               Krystyna Ohnesorge, Swiss Federal Archives, Berne, Switzerland
JAVA         : Version 1.8.0_212 (64)
Linux        : Version 2.6.32-696.10.1.el6.x86_64 (amd64)
Sep 26, 2019 1:09:53 PM ch.enterag.utils.logging.IndentLogger log
INFO: SiardToDb null - Program to load database content from a .siard file
Sep 26, 2019 1:09:53 PM ch.enterag.utils.logging.IndentLogger log
INFO: SIARD Suite null: Swiss Federal Archives, Berne, Switzerland, 2008-2016
Sep 26, 2019 1:09:53 PM ch.enterag.utils.logging.IndentLogger log
INFO: Specified by : Hartwig Thomas, Enter AG, R?ti ZH, Switzerland
Sep 26, 2019 1:09:53 PM ch.enterag.utils.logging.IndentLogger log
INFO:                Andreas Voss, Swiss Federal Archives, Berne, Switzerland
Sep 26, 2019 1:09:53 PM ch.enterag.utils.logging.IndentLogger log
INFO:                Anders Bo Nielsen, Danish National Archives, Denmark
Sep 26, 2019 1:09:53 PM ch.enterag.utils.logging.IndentLogger log
INFO:                Claire R?thlisberger-Jourdan, KOST, Berne, Switzerland
Sep 26, 2019 1:09:53 PM ch.enterag.utils.logging.IndentLogger log
INFO: Developed by : Hartwig Thomas, Enter AG, R?ti ZH, Switzerland
Sep 26, 2019 1:09:53 PM ch.enterag.utils.logging.IndentLogger log
INFO:                Simon Jutz, Cytex GmbH, Zurich, Switzerland
Sep 26, 2019 1:09:53 PM ch.enterag.utils.logging.IndentLogger log
INFO: Tested by    : Claudia Matthys, POOL Computer AG, Zurich, Switzerland
Sep 26, 2019 1:09:53 PM ch.enterag.utils.logging.IndentLogger log
INFO:                Marcel B?chler, Swiss Federal Archives, Berne, Switzerland
Sep 26, 2019 1:09:53 PM ch.enterag.utils.logging.IndentLogger log
INFO:                Yvan Dutoit, Swiss Federal Archives, Berne, Switzerland
Sep 26, 2019 1:09:53 PM ch.enterag.utils.logging.IndentLogger log
INFO: Managed by   : Hartwig Thomas, Enter AG, R?ti ZH, Switzerland
```

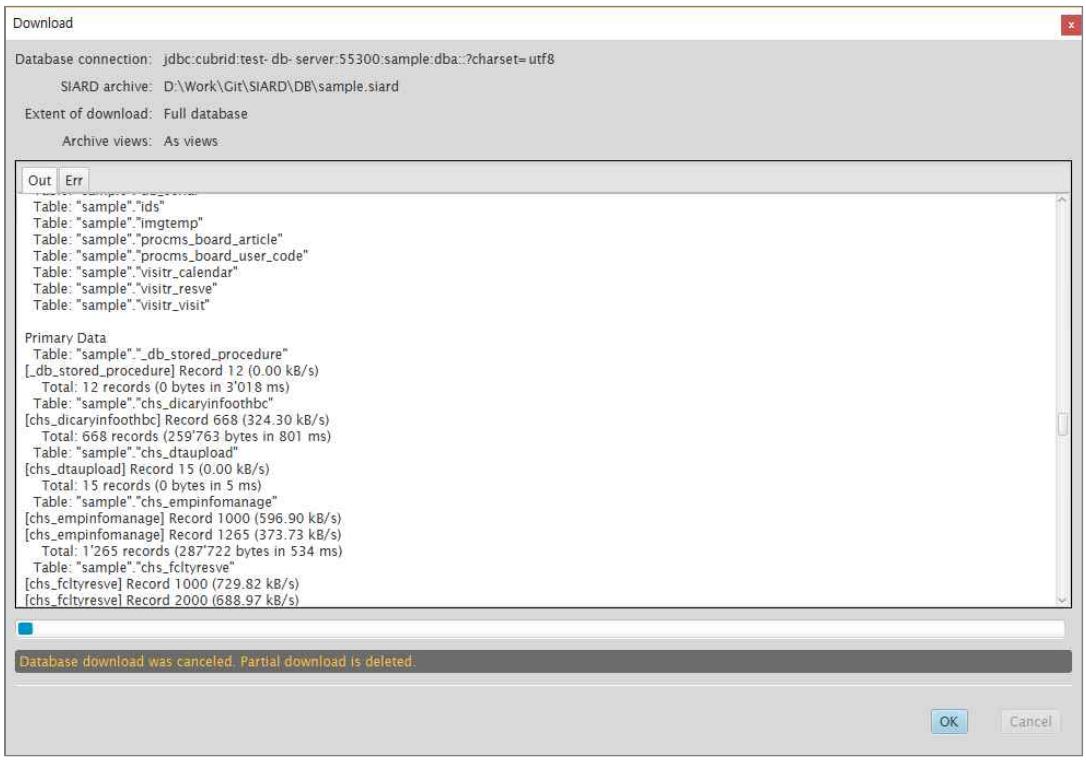
<그림72> SiardCmd-ToDB.jar 실행 화면 : Upload (SIARD파일 → DBMS)

```
Sep 26, 2019 1:09:53 PM ch.enterag.utils.logging.IndentLogger log
INFO:   Database user          : dba
  Database user          : dba
Sep 26, 2019 1:09:53 PM ch.enterag.utils.logging.IndentLogger log
INFO:   Database password      : ***
  Database password      : ***
Sep 26, 2019 1:09:53 PM ch.enterag.utils.logging.IndentLogger log
INFO:

Sep 26, 2019 1:09:54 PM com.sun.xml.bind.v2.runtime.reflect.opt.AccessorInjector <clinit>
INFO: The optimized code generation is disabled
Connected to jdbc:cubrid:localhost:55300:chungsa:dba::?charset=utf8
Meta Data
  Table: "sample"."chs_dicaryinfoothbc"
  Table: "sample"."chs_dtaupload"
  Table: "sample"."chs_empinfomanage"
  Table: "sample"."chs_fcltyresve"
  Table: "sample"."chs_fcltyresvegrp"
  Table: "sample"."chs_fcltyresvember"
  Table: "sample"."chs_fcltyrsvdeban"
  Table: "sample"."chs_gbdacctomvinstt"
  Table: "sample"."chs_gbdaptcmplacctomainengn"
  Table: "sample"."chs_gbdflooracctodept"
  Table: "sample"."chs_gbdflooracctodept_bak"
  Table: "sample"."chs_gbdflooracctosumry"
  Table: "sample"."chs_infoothbc"
  Table: "sample"."chs_popupplacemanage"
  Table: "sample"."chs_rstrntmenumanage"
  Table: "sample"."chungsa_page_menu_info"
  Table: "sample"."chungsa_page_view_log"
  Table: "sample"."chungsa_page_view_log_back"
  Table: "sample"."chungsa_page_view_summary"
  Table: "sample"."comtcadministcode"
  Table: "sample"."comtcadministcoderecptnlog"
  Table: "sample"."comtccmmnclcode"
  Table: "sample"."comtccmmncode"
  Table: "sample"."comtccmmndetailcode"
  Table: "sample"."comtczip"
  Table: "sample"."comtecopseq"
  Table: "sample"."comthconfmhistory"
  Table: "sample"."comthdbmntrngloginfo"
  Table: "sample"."comthemaildsptchmanage"
  Table: "sample"."visitr_resve"
  Table: "sample"."visitr_visit"

Primary Data
  Table: "sample"."_db_stored_procedure"
    Record 0 (0.00 kB/s)
    Total: 0 records (259 bytes in 0 ms)
  Table: "sample"."chs_dicaryinfoothbc"
    Record 668 (157.74 kB/s)
    Total: 668 records (260'535 bytes in 1'601 ms)
  Table: "sample"."chs_dtaupload"
    Record 15 (0.00 kB/s)
    Total: 15 records (3'248 bytes in 20 ms)
  Table: "sample"."chs_empinfomanage"
    Record 1000 (129.59 kB/s)
    Record 1265 (126.66 kB/s)
    Total: 1'265 records (290'491 bytes in 2'188 ms)
  Table: "sample"."chs_fcltyresve"
    Record 1000 (177.35 kB/s)
    Record 2000 (149.33 kB/s)
    Record 3000 (177.72 kB/s)
    Record 4000 (180.65 kB/s)
    Record 5000 (155.62 kB/s)
    Record 6000 (184.68 kB/s)
    Record 7000 (187.27 kB/s)
    Record 7956 (170.54 kB/s)
    Total: 7'956 records (1'960'008 bytes in 11'360 ms)
  Table: "sample"."chs_fcltyresvegrp"
    Record 1000 (186.73 kB/s)
    Record 1588 (134.98 kB/s)
    Total: 1'588 records (483'190 bytes in 2'904 ms)
  Table: "sample"."chs_fcltyresvember"
    Record 1000 (117.32 kB/s)
    Record 2000 (139.10 kB/s)
    Record 2943 (128.70 kB/s)
    Total: 2'943 records (619'398 bytes in 4'784 ms)
  Table: "sample"."chs_fcltyrsvdeban"
    Record 1000 (135.45 kB/s)
    Record 2000 (193.31 kB/s)
    Record 3000 (179.84 kB/s)
    Record 3532 (161.07 kB/s)
    Total: 3'532 records (990'347 bytes in 5'962 ms)
```

<그림 73> SiardCmd-ToDB.jar 실행 화면 Upload (SIARD파일 → DBMS)



<그림 74> SiardGui 실행 화면 : Download (DBMS → SIARD파일)



<그림75> SiardGui 실행 화면 : Download (DBMS → SIARD파일)

5.2 Oracle DB 대상 실데이터 검증

5.2.1 실데이터 검증 방법

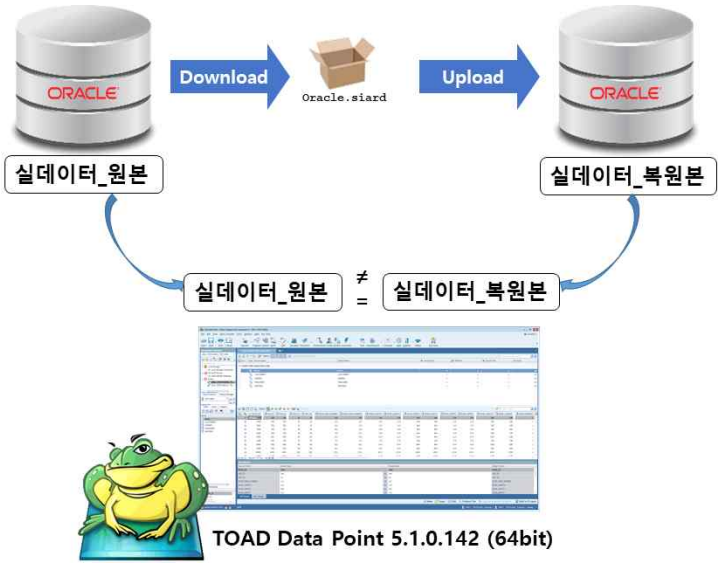
○ 검증 대상 DBMS 내역

구분	내용
DBMS 종류 및 버전	Oracle 11g R2
원본 DB 크기	1550.9 MB
테이블 개수(개) 및 레코드 수(건)	73개 Table, 총 12,479,204 건
Download/Upload 도구	SiardGui for 큐브리드
비교 검증 도구	SQL Developer, SiardGui, TOAD Data Point 5.1.0.142

<표 93> 검증 대상 DBMS 내역

○ 실데이터 검증 방법

- Oracle DBMS에 접속하여 원본DB를 Download하여 SIARD 파일 생성받고, 다시 생성된 SIARD 파일을 Oracle DBMS에 접속하여 업로드함
- SQL Developer, SiardGui를 이용하여 원본DB, SIARD, 업로드DB의 테이블명, 레코드 건수를 비교
- TOAD Data Point 5.1.0으로 두 개의 데이터베이스에 접속하여 두 개 데이터베이스의 데이터에 대하여 Data Diff Viewer 기능을 통해 데이터 비교



<그림 76> 실데이터 검증 방법

5.2.2 실데이터 검증 결과

○ 테이블 및 데이터 건수 비교(73개 모든 테이블에 대해)

- SQL Developer 및 SiardGui를 이용하여 원본DB, SIARD 파일, 업로드DB의 테이블명 및 건수 확인

- 원본DB, SIARD 파일, 업로드DB의 모든 테이블명, TB_COURSES_USERS를 제외하고 건수 모두 동일함(<표 94> 참고)

원본DB 테이블명	업로드DB 테이블명	비교	원본DB(건)	SIARD(건)	업로드DB(건)
PRV_INFO_ACCS_HIST	PRV_INFO_ACCS_HIST	동일	38,477	38,477	38,477
TB_ATTENDANCES	TB_ATTENDANCES	동일	17,820	17,820	17,820
TB_ATTENDANCES_BAK20150610	TB_ATTENDANCES_BAK20150610	동일	4	4	4
TB_BOARDS	TB_BOARDS	동일	6	6	6
TB_CATEGORIES	TB_CATEGORIES	동일	-	-	-
TB_CODES	TB_CODES	동일	9,702	9,702	9,702
TB_CODES_20141031BAK	TB_CODES_20141031BAK	동일	8,752	8,752	8,752
TB_CODES_IMSI	TB_CODES_IMSI	동일	9,702	9,702	9,702
TB_COMMENTS	TB_COMMENTS	동일	-	-	-
TB_CONTENTS	TB_CONTENTS	동일	8	8	8
TB_CONTENTS_BAK20150610	TB_CONTENTS_BAK20150610	동일	4	4	4
TB_CONTENT_ITEMS	TB_CONTENT_ITEMS	동일	-	-	-
TB_CONTENT_META	TB_CONTENT_META	동일	5		5
TB_CONTENT_RESOURCES	TB_CONTENT_RESOURCES	동일	-	-	-
TB_COURSES	TB_COURSES	동일	403	403	403
TB_COURSE_LECTURES	TB_COURSE_LECTURES	동일	2,570	2,570	2,570
TB_COURSE_SCHEDULES	TB_COURSE_SCHEDULES	동일	5,704	5,704	5,704
TB_COURSE_USERS	TB_COURSE_USERS	동일	12,124	12,124	12,124
TB_EDUCATIONS	TB_EDUCATIONS	동일	196	196	196
TB_EXAMS	TB_EXAMS	동일	260	260	260
TB_EXAM_SCORES	TB_EXAM_SCORES	동일	138,763	138,763	138,763
TB_EXAM_USERS	TB_EXAM_USERS	동일	6,939	6,939	6,939
TB_FILES	TB_FILES	동일	785	785	785
TB_LECTURES	TB_LECTURES	동일	1,407	1,407	1,407
TB_LESSONS	TB_LESSONS	동일	609	609	609
TB_MAILINGS	TB_MAILINGS	동일	-	-	-
TB_MAILING_LOG	TB_MAILING_LOG	동일	-	-	-
TB_MESSAGES	TB_MESSAGES	동일	-	-	-
TB_MESSAGE_LOG	TB_MESSAGE_LOG	동일	-	-	-
TB_METADATAS	TB_METADATAS	동일	61	61	61
TB_ORGANIZATIONS	TB_ORGANIZATIONS	동일	368,138	368,138	368,138

TB_ORGANIZATIONS_20130328	TB_ORGANIZATIONS_20130328	동일		113,755	113,755	113,755
TB_ORGANIZATIONS_20141031BAK	TB_ORGANIZATIONS_20141031BAK	동일		112,131	112,131	112,131
TB_ORGANIZATIONS_20141205	TB_ORGANIZATIONS_20141205	동일		309,038	309,038	309,038
TB_ORGANIZATIONS_20150120BAK	TB_ORGANIZATIONS_20150120BAK	동일		309,038	309,038	309,038
TB_ORGANIZATIONS_20151215BAK	TB_ORGANIZATIONS_20151215BAK	동일		324,377	324,377	324,377
TB_ORGANIZATIONS_20170411	TB_ORGANIZATIONS_20170411	동일		330,061	330,061	330,061
TB_ORGANIZATIONS_20190319	TB_ORGANIZATIONS_20190319	동일		352,123	352,123	352,123
TB_ORGANIZATIONS_BAK20130401	TB_ORGANIZATIONS_BAK20130401	동일		113,755	113,755	113,755
TB_ORGANIZATIONS_IMSI	TB_ORGANIZATIONS_IMSI	동일		368,137	368,137	368,137
TB_ORGANIZATIONS_NEW	TB_ORGANIZATIONS_NEW	동일		112,128	112,128	112,128
TB_PAGE_MANAGER	TB_PAGE_MANAGER	동일		22	22	22
TB_PAGE_MANAGER_BACK	TB_PAGE_MANAGER_BACK	동일		69	69	69
TB_POLLS	TB_POLLS	동일		2,377	2,377	2,377
TB_POLLS_20151221BAK	TB_POLLS_20151221BAK	동일		1,586	1,586	1,586
TB_POLLS_BAK20150609	TB_POLLS_BAK20150609	동일		12	12	12
TB_POLL_ANSWERS	TB_POLL_ANSWERS	동일		231,314	231,314	231,314
TB_POLL_ANSWERS_20141217BAK	TB_POLL_ANSWERS_20141217BAK	동일		9	9	9
TB_POLL_ANSWERS_20151221BAK	TB_POLL_ANSWERS_20151221BAK	동일		162,278	162,278	162,278
TB_POLL_ITEMS	TB_POLL_ITEMS	동일		13	13	13
TB_POPUPS	TB_POPUPS	동일		27	27	27
TB_PROFESSORS	TB_PROFESSORS	동일		41	41	41
TB_QNA	TB_QNA	동일		-	-	-
TB_QUESTIONS	TB_QUESTIONS	동일		541	541	541
TB_RESEARCHES	TB_RESEARCHES	동일		4	4	4
TB_RESEARCH_ANSWERS	TB_RESEARCH_ANSWERS	동일		4	4	4
TB_RESULT	TB_RESULT	동일		416	416	416
TB_SMS	TB_SMS	동일		-	-	-
TB_SMS_LOG	TB_SMS_LOG	동일		-	-	-
TB_STATISTIC_LOG	TB_STATISTIC_LOG	동일		2,461,440	2,461,440	2,461,440
TB_SURVEY	TB_SURVEY	동일		71	71	71
TB_SURVEY_EVENT	TB_SURVEY_EVENT	동일		11	11	11
TB_SURVEY_QUESTION	TB_SURVEY_QUESTION	동일		323	323	323
TB_SURVEY_QUESTION_ITEM	TB_SURVEY_QUESTION_ITEM	동일		1,029	1,029	1,029
TB_SURVEY_RESULT	TB_SURVEY_RESULT	동일		5,429	5,429	5,429
TB_TEACHERS	TB_TEACHERS	동일		9	9	9
TB_USERS	TB_USERS	동일		12,122	12,122	12,122
TB_USERS_BAK20130401	TB_USERS_BAK20130401	동일		6,279	6,279	6,279
TB_USERS_NEW	TB_USERS_NEW	동일		6,279	6,279	6,279
TB_WEBLOG	TB_WEBLOG	동일		360,434	360,434	360,434
TB_WEBLOG_REFERERS	TB_WEBLOG_REFERERS	동일		65,428	65,428	65,428
TB_ZIPCODE	TB_ZIPCODE	동일		6,044,302	6,044,302	6,044,302
TB_ZIPCODE_OLD	TB_ZIPCODE_OLD	동일		50,353	50,353	50,353

<표 94> 실데이터 건수 검증 결과

○ 데이터 비교(백업용 제외하고 73개 모든 테이블에 대해)

- TOAD Data Point 5.1.0.142을 활용하여 원본DB와 업로드DB의 모든 레코드의 데이터 비교 진행

○ Oracle DATE 타입을 포함하고 있는 일부 테이블의 레코드들에서 다른 데이터가 있음을 확인함(<표 95> 참고)

원본DB 테이블명	업로드DB 테이블명	다른 레코드 수 (건)	동일한 레코드 수 (건)
PRV_INFO_ACCS_HIST	PRV_INFO_ACCS_HIST	38,477	0
TB_ATTENDANCES	TB_ATTENDANCES	0	17,820
TB_ATTENDANCES_BAK20150610	TB_ATTENDANCES_BAK20150610	0	4
TB_BOARDS	TB_BOARDS	0	6
TB_CATEGORIES	TB_CATEGORIES	0	0
TB_CODES	TB_CODES	0	9,702
TB_CODES_20141031BAK	TB_CODES_20141031BAK	0	8,752
TB_CODES_IMSI	TB_CODES_IMSI	0	9,702
TB_COMMENTS	TB_COMMENTS	0	0
TB_CONTENTS	TB_CONTENTS	0	8
TB_CONTENTS_BAK20150610	TB_CONTENTS_BAK20150610	0	4
TB_CONTENT_ITEMS	TB_CONTENT_ITEMS	0	0
TB_CONTENT_META	TB_CONTENT_META	0	5
TB_CONTENT_RESOURCES	TB_CONTENT_RESOURCES	0	0
TB_COURSES	TB_COURSES	0	403
TB_COURSE_LECTURES	TB_COURSE_LECTURES	0	2,570
TB_COURSE_SCHEDULES	TB_COURSE_SCHEDULES	0	5,704
TB_COURSE_USERS	TB_COURSE_USERS	0	12,414
TB_EDUCATIONS	TB_EDUCATIONS	0	196
TB_EXAMS	TB_EXAMS	0	260
TB_EXAM_SCORES	TB_EXAM_SCORES	0	138,763
TB_EXAM_USERS	TB_EXAM_USERS	0	6,939
TB_FILES	TB_FILES	0	785
TB_LECTURES	TB_LECTURES	0	1,407
TB_LESSONS	TB_LESSONS	0	609
TB_MAILINGS	TB_MAILINGS	0	0
TB_MAILING_LOG	TB_MAILING_LOG	0	0
TB_MESSAGES	TB_MESSAGES	0	0
TB_MESSAGE_LOG	TB_MESSAGE_LOG	0	0
TB_METADATAS	TB_METADATAS	0	61
TB_ORGANIZATIONS	TB_ORGANIZATIONS	0	368,138

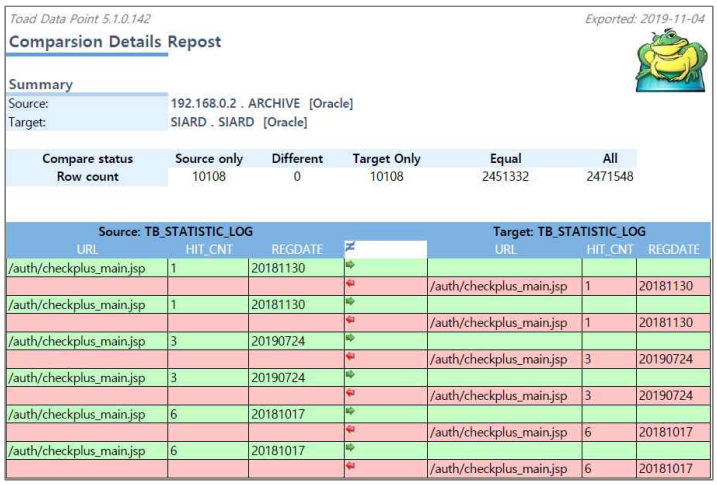
TB_ORGANIZATIONS_20130328	TB_ORGANIZATIONS_20130328	0	113,755
TB_ORGANIZATIONS_20141031BAK	TB_ORGANIZATIONS_20141031BAK	0	112,131
TB_ORGANIZATIONS_20141205	TB_ORGANIZATIONS_20141205	0	309,038
TB_ORGANIZATIONS_20150120BAK	TB_ORGANIZATIONS_20150120BAK	0	309,038
TB_ORGANIZATIONS_20151215BAK	TB_ORGANIZATIONS_20151215BAK	0	324,377
TB_ORGANIZATIONS_20170411	TB_ORGANIZATIONS_20170411	0	330,061
TB_ORGANIZATIONS_20190319	TB_ORGANIZATIONS_20190319	0	352,123
TB_ORGANIZATIONS_BAK20130401	TB_ORGANIZATIONS_BAK20130401	0	113,755
TB_ORGANIZATIONS_IMSI	TB_ORGANIZATIONS_IMSI	0	368,137
TB_ORGANIZATIONS_NEW	TB_ORGANIZATIONS_NEW	0	112,128
TB_PAGE_MANAGER	TB_PAGE_MANAGER	0	22
TB_PAGE_MANAGER_BACK	TB_PAGE_MANAGER_BACK	0	69
TB_POLLS	TB_POLLS	0	2,377
TB_POLLS_20151221BAK	TB_POLLS_20151221BAK	0	1,568
TB_POLLS_BAK20150609	TB_POLLS_BAK20150609	0	12
TB_POLL_ANSWERS	TB_POLL_ANSWERS	0	231,314
TB_POLL_ANSWERS_20141217BAK	TB_POLL_ANSWERS_20141217BAK	0	22
TB_POLL_ANSWERS_20151221BAK	TB_POLL_ANSWERS_20151221BAK	0	162,278
TB_POLL_ITEMS	TB_POLL_ITEMS	0	13
TB_POPUPS	TB_POPUPS	0	27
TB_PROFESSORS	TB_PROFESSORS	0	41
TB_QNA	TB_QNA	0	0
TB_QUESTIONS	TB_QUESTIONS	0	541
TB_RESEARCHES	TB_RESEARCHES	0	4
TB_RESEARCH_ANSWERS	TB_RESEARCH_ANSWERS	0	4
TB_RESULT	TB_RESULT	0	416
TB_SMS	TB_SMS	0	0
TB_SMS_LOG	TB_SMS_LOG	0	0
TB_STATISTIC_LOG	TB_STATISTIC_LOG	0	2,461,440
TB_SURVEY	TB_SURVEY	0	71
TB_SURVEY_EVENT	TB_SURVEY_EVENT	11	0
TB_SURVEY_QUESTION	TB_SURVEY_QUESTION	323	0
TB_SURVEY_QUESTION_ITEM	TB_SURVEY_QUESTION_ITEM	1,029	0
TB_SURVEY_RESULT	TB_SURVEY_RESULT	5,429	0
TB_TEACHERS	TB_TEACHERS	0	9
TB_USERS	TB_USERS	17	12,105
TB_USERS_BAK20130401	TB_USERS_BAK20130401	0	6,279
TB_USERS_NEW	TB_USERS_NEW	0	6,279
TB_WEBLOG	TB_WEBLOG	0	360,434
TB_WEBLOG_REFERERS	TB_WEBLOG_REFERERS	0	65,428
TB_ZIPCODE	TB_ZIPCODE	0	6,044,302
TB_ZIPCODE_OLD	TB_ZIPCODE_OLD	0	50,353

<표 95> 실데이터 데이터 검증 결과

- Oracle DATE 타입을 포함하고 있는 일부 테이블의 레코드들에서 원본DB, SIARD 파일, 업로드DB가 서로 데이터가 다른 레코드가 있다는 것을 확인함(<표 95> 참고)

- 오류 발생 현상 및 예시, 원인은 <표 96>에서 확인할 수 있음

<table>
<tr><th>유형</th><th>구분</th><th colspan="2">분석결과</th></tr>
<tr><td rowspan="3">1</td><td rowspan="3">PRV_INFO_ACCS_HIST,
TB_SURVEY_EVENT,
TB_SURVEY_QUESTION,
TB_SURVEY_QUESTION_ITEM,
TB_SURVEY_RESULT
총 5개 테이블
(“다른 레코드 수”만 있는 테이블)</td><td>현상</td><td>· Oracle에서 DATE 타입으로 선언되어 있는 컬럼 데이터가 다름
(원본DB ≠ SIARD 파일 = 업로드DB)</td></tr>
<tr><td>예시</td><td>
<table>
<tr><th colspan="2">원본DB: TB_SURVEY_EVENT</th></tr>
<tr><th>FST_REG_DT</th><th>LST_UPD_DT</th></tr>
<tr><td>2017-11-20
오전 10:19:33</td><td>2017-11-20
오전 10:19:33</td></tr>
<tr><td>2017-11-20
오전 10:19:34</td><td>2017-11-20
오전 10:19:34</td></tr>
<tr><td>2018-11-07
오전 10:44:58</td><td>2018-11-07
오전 10:44:58</td></tr>
</table>
≠
<table>
<tr><th colspan="2">업로드DB: TB_SURVEY_EVENT</th></tr>
<tr><th>FST_REG_DT</th><th>LST_UPD_DT</th></tr>
<tr><td>2017-11-20
오전 12:00:00</td><td>2017-11-20
오전 12:00:00</td></tr>
<tr><td>2017-11-20
오전 12:00:00</td><td>2017-11-20
오전 12:00:00</td></tr>
<tr><td>2018-11-07
오전 12:00:00</td><td>2018-11-07
오전 12:00:00</td></tr>
</table>
</td></tr>
<tr><td>원인</td><td>· Oracle DATE 타입은 “년/월/일”과 “시/분/초”까지 저장함
· 반면, 오프소스 프로젝트 SIARD Suite에서 DATE 타입을 가져올 때 JDBC API 중 “Date getDate(int columnIndex) throws SQLException”를 사용하고 있는데, getDate(..)는 “년/월/일”만 가져 오도록 되어 있기 때문에 ”시/분/초“의 데이터가 사라지게 되어 ‘원본DB 레코드’와 ‘SIARD 파일(업로드DB) 레코드’의 데이터 값이 다르게 되어 있음</td></tr>
<tr><td rowspan="3">2</td><td rowspan="3">TB_USERS
(“다른 레코드 수“와 “동일한 레코드 수“가 존재하는 있는 테이블)</td><td>현상</td><td>· 다른 레코드 수(17건)은 Oracle DATE 타입이 있는 레코드
(원본DB ≠ SIARD 파일 = 업로드DB)
· 동일한 레코드 수(12,105건)은 Oracle에서는 DATE 타입으로 선언되어 있지만 값이 null이므로 동일한 레코드가 된 경우임
(원본DB ≠ SIARD 파일 = 업로드DB)</td></tr>
<tr><td>예시</td><td>
<table>
<tr><th colspan="1">원본DB: TB_USERS</th></tr>
<tr><th>LST_LOGIN_DT</th></tr>
<tr><td>2019-08-30 오전 11:19:56</td></tr>
<tr><td>2019-06-12 오후 3:35:43</td></tr>
<tr><td>2019-09-06 오후 2:33:45</td></tr>
</table>
≠
<table>
<tr><th colspan="1">업로드DB: TB_USERS</th></tr>
<tr><th>LST_LOGIN_DT</th></tr>
<tr><td>2019-08-30 오전 12:00:00</td></tr>
<tr><td>2019-06-12 오전 12:00:00</td></tr>
<tr><td>2019-09-06 오전 12:00:00</td></tr>
</table>
<table>
<tr><th colspan="1">원본DB: TB_USERS</th></tr>
<tr><th>LST_LOGIN_DT</th></tr>
<tr><td>null</td></tr>
<tr><td>null</td></tr>
</table>
=
<table>
<tr><th colspan="1">업로드DB: TB_USERS</th></tr>
<tr><th>LST_LOGIN_DT</th></tr>
<tr><td>null</td></tr>
<tr><td>null</td></tr>
</table>
</td></tr>
<tr><td>원인</td><td>· DATE 값이 설정되어 있는 경우는 유형1과 동일한 원인으로 인한 것이며, DATE 값이 설정되지 않아 null인 경우는 업로드DB로 null로 설정되어 있어 동일한 레코드로 인식</td></tr>
</table>

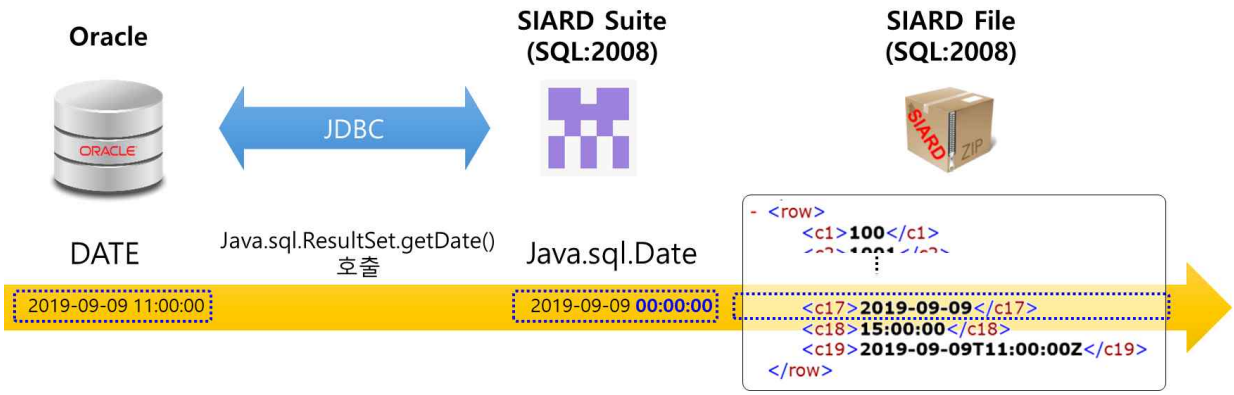
3	TB_STATISTIC_LOG	· TOAD Data Point에서 비교한 2,461,440건 레코드 중에서 동일하지 않은 데이터가 원본DB와 업로드DB에 각각 10,108건 따로 존재한다고 보고되었지만,실제 데이터를 육안으로 비교해 보니 동일한 데이터라고 판단됨  Toad Data Point 5.1.0.142 Exported: 2019-11-04 Comparsion Details Repost Summary Source: 192.168.0.2 . ARCHIVE [Oracle] Target: SIARD . SIARD [Oracle] Compare status / Row count: Source only 10108, Different 0, Target Only 10108, Equal 2451332, All 2471548 Source: TB_STATISTIC_LOG / Target: TB_STATISTIC_LOG URL HIT_CNT REGDATE /auth/checkplus_main.jsp 1 20181130 /auth/checkplus_main.jsp 1 20181130 /auth/checkplus_main.jsp 3 20190724 /auth/checkplus_main.jsp 3 20190724 /auth/checkplus_main.jsp 6 20181017 /auth/checkplus_main.jsp 6 20181017

<표 96> 실데이터 비교 분석

5.2.3 문제점 해결 방안

○ 데이터가 다른 레코드 해결방안

- SIARD 2.1 표준을 SFA(Swiss Federal Archives)에서 구현한 오픈소스 프로젝트 SIARD Suite은 기본적으로 JDBC를 사용하고 DBMS와 연결하고 있음
- Oracle의 DATE 타입은 SQL:2008의 DATE 타입으로 대응되어 있음
- 그러므로, JDBC API 중에서 java.sql.getDate(Date getDate(int columnIndex) throws SQLException)로 DBMS로부터 데이터를 가지고 오도록 구현되어 있음



<그림 77> SIARD Suite에서의 Oracle DATE 타입의 SIARD 파일 변환과정

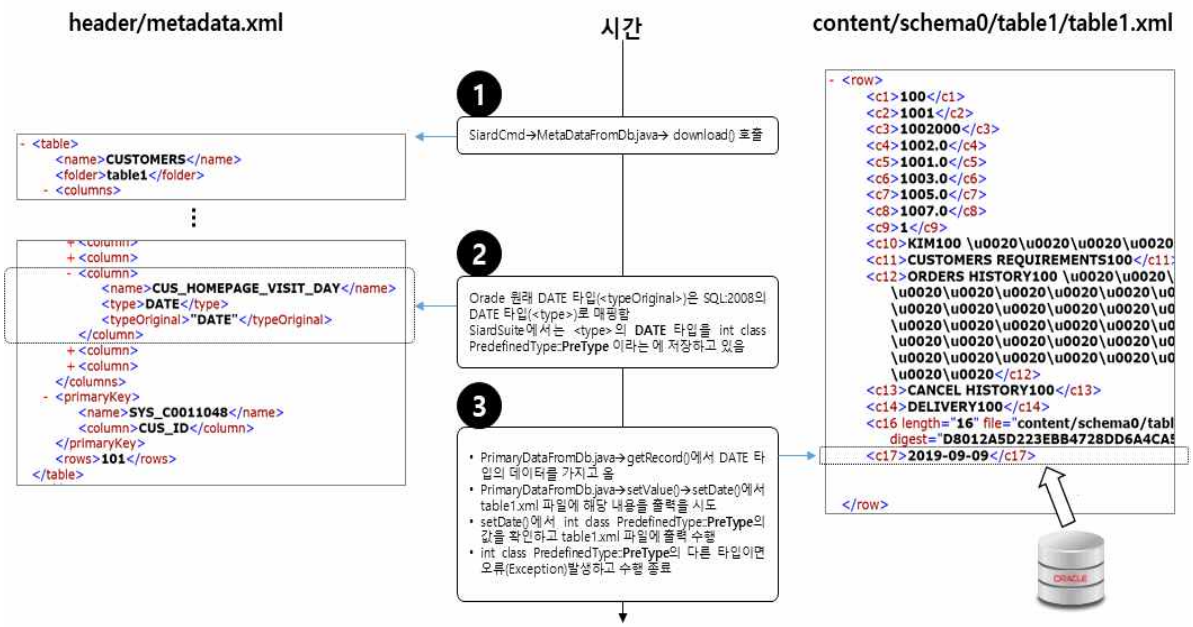
- 이때, 반환되는 Date 타입(java.sql.Date)에는 년/월/일은 있지만 시/분/초는 가져오지 않고 있음
  (※ getDate(...)함수는 SIARD Suite → SiardCmd 프로젝트 → src → PrimaryDataFromDb.java → private void getRecord(ResultSet rs, Record record) throws IOException, SQLException 에서 호출됨)

- 최종 SIARD 파일로 저장될 때에도 시/분/초 값이 없는 상태로 저장됨

- java.sql.Date 타입을 Oracle DB에서 가지고 올 때 getTimestamp(Timestamp getTimestamp(int columnIndex) throws SQLException)로 가지고 올 수 있기 때문에, Oracle의 경우 상황에 따라 getDate(...)와 getTimestamp(...)로 가지고 올 수 있도록 SIARD Suite의 일부 소스코드를 수정 및 보완되어야 함

- 단, Oracle의 경우에만 단순히 getDate(...) 대신 getTimestamp(...)로 가지고 오도록 수정하면 오류가 발생함

- 소스코드에서는 <그림 78>과 같이 처리되고 있으며, SIARD Suite에서는 각 데이터베이스의 타입들에 대한 SQL:2008에 대응되는 타입을 미리 정해 놓고, 해당되는 타입을 가지고 오는 JDBC API 함수도 결정되어 있는 상황임. Oracle의 DATE 타입의 경우는 SQL:2008의 DATE 타입으로 정해졌으며, JDBC API에서는 java.sql.getDate() 함수로 가져오게 되어 있음



<그림 78> Oracle DATE 타입에 대한 SIARD Suite 소스코드에서의 처리과정

- ❶ SiardCmd → MetaDataFromDb.java → download() 가 호출되면 metadata.xml이 생성되고 데이터베이스 내에 테이블에 대한 전체 구조에 대한 내용이 저장됨

- ❷ 반드시 Oracle의 DATE 타입(metadata.xml에서는 <typeOriginal>로 표현)은 SQL:2008의 DATE 타입(metadata.xml에서는 <type>으로 표현)으로 정해져 있고, PreType:java.sql.getDate(...)함수를 실행하기 하고 추후 getDate(...)함수가 잘 수행되었는지 여부를 확인하기 위해 int class PredefinedType::PreType에 DATE 타입으로 저장함

- ❸ Oracle DATE 타입을 PrimaryDataFromDb.java → getRecord(...) → getDate(...)로 DB에서 가지고 온 다음 PrimaryDataFromDb.java → setValue(...) → setDate(...)를 실행함. PrimaryDataFromDb.java → setValue(...) → setDate(...)은 SFA SIARD Suite 오픈소스의 SiardApi 프로젝트의 ValueImple.java에 구현되어 있으며, 실제 파일에 쓰기 전에 int class PredefinedType::PreType에 저장되어 있는 DATE 타입과 동일한 타입인지 검사함

- 만약에 Oracle DATE 타입에 대해 java.sql.getDate(...) 함수가 아닌 java.sql.getTimestamp(...)함수로 실행된다면, DB로부터 가지고 온 타입이 Timestamp 타입으로 바뀌기 때문에 오류를 발생시키고 프로그램을 강제 종료됨

- 그러므로 Oracle DATE 타입의 시/분/초까지 모두 SIARD 파일로 저장하기 위해서, <표 97>처럼 다음의 세 가지 구현 방안을 제안하고, **방안 1**을 구현하여 소스코드에 반영함

연번	구분	구현 방안	
방안1	SIARD Suite의 SiardCmd 프로젝트 소스코드 수정	수정코드	· SiardCmd의 PrimaryDataFromDb.java 수정
		수정전	... case Types.DATE: oValue = rs.getDate(iPosition); break; ...
		수정후	... case Types.DATE: **if (_dbms.equals("Oracle")) {** **mc.setPreType(Types.TIMESTAMP, 0, mc.getScale());** **oValue = rs.getTimestamp(iPosition);** **}** **else {** oValue = rs.getDate(iPosition); **}** break; ...
		결과	· 모든 레코드 동일함

방안2	Oracle JDBC 소스코드 수정하는 방안	· SIARD Suite의 JdbcOracle 프로젝트는 Oracle JDBC Driver의 Wrapper 함수로 JDBC API를 SIARD Suite의 다른 프로젝트에서 일관된 방식으로 호출할 수 있도록 설계되어 있음 · Oracle JDBC API중 java.sql.ResultSet.getDate(...)함수가 년/월/일과 시/분/초 모두 가져올 수 있도록 하기 위해서는 Oracle JDBC Driver 소스코드 자체를 수정해야 함 · (주의사항) JDBC API의 동작과정을 수정하는 것은 JDBC라는 모든 DBMS에 대해 일반화(Normalization)된 구조에 위배되는 것으로 추후 Oracle JDBC에 대해 호환성에 문제가 발생할 수 있음
방안3	SIARD Suite의 SqlParser 프로젝트 소스코드 수정	· SIARD Suite에서는 각 DBMS에서 제공하고 있는 타입들에 대해 SQL:2008의 타입들도 대응시키는 규칙을 가지고 있음 · 여기에서 DATE 타입이 TIMESTAMP 타입과 호환이 되도록 코드를 수정 방안임 · (주의사항) SqlParser 프로젝트는 SIARD Suite 중에서 가장 복잡하면서 정교하게 작업된 프로젝트로 소스코드 분석에 많은 시간과 노력이 예상됨

<표 97> Oracle DATE 타입에 대한 구현 방안

6. 테스트베드 구축 · 시험 결과에 따른 장기보존방안

- 큐브리드 지원하기 위해 SIARD Suite를 확장 및 개발하고, 4개 DBMS에 대한 검증 수행한 결과를 분석하여 장기보존방안을 제시함
- 먼저, SIARD Suite이 가지고 있는 장단점을 분석하고, 이를 기반으로 향후 SIARD 표준과 SIARD Suite을 이용하여 데이터세트 보존에 활용하는 방안을 제안함
- 또한, 향후 SIARD Suite를 확장하여 다른 DBMS를 지원하고자 할 때, 어떤 단계로 진행해야 하는지, 어느 부분을 수정 및 보완해야 하는지 제시함

6.1 SIARD 표준을 활용하기 위해 표준 및 오픈소스 장단점

- SIARD Suite은 SIARD 표준에 따라 다양한 DBMS에 저장되어 있는 요소들을 SIARD 표준에 맞춰 SIARD 파일로 변환시키고 복원까지 수행할 수 있는 소프트웨어이지만, 현재 모든 DBMS를 지원하지 않기 때문에 추가적인 분석 및 개발을 필요함

6.1.1 SIARD 2.1 및 SIARD Suite 장점

(1) 오픈소스로 무료로 사용할 수 있음
(https://github.com/sfa-siard/SiardGui/releases)

(2) SIARD Suite은 CDDL 오픈소스 라이선스로 별도로 작성한 소스코드의 저작권을 보장할 수 있음

(3) 현재 대표적인 DBMS 관계형 DBMS 6개(Oracle, MS SQL Server, MySQL, DB/2, MS Access)를 지원하고 있음

(4) DBMS→SIARD파일(변환), SIARD파일→DBMS(복원) 모두 가능함

(5) 또한, A라는 DBMS로부터 변환된 SIARD파일은 B라는 다른 DBMS로도 복원이 가능함

(6) SIARD Suite은 독자적인 표준이 아닌 SQL:2008, Unicode, XML, ZIP64와 같은 국제적으로 공인받은 표준들 토대로 제정된 표준인 SIARD 2.1을 기반으로 구현되었기 때문에 DB형 데이터세트의 “장기보존”에 적합함

(7) SIARD Suite은 JAVA 언어로 구현되어서 플랫폼(OS)마다 별도의 소스코드를 작성할 필요없이 하나의 소스코드만 작성하더라도 다양한 플랫폼에서 동일하게 실행할 수 있음

(8) SIARD Suite은 JAVA 언어로 구현되었기 때문에, DBMS에 접속하여 일련의 데이터 처리 과정을 수행할 때 JDBC API를 사용하는데, JDBC API는 대부분의 DBMS에서 기본적으로 제공하기 때문에 현재 6개 이외에 다른 DBMS도 SIARD Suite의 지원 DBMS 목록에 포함될 수 있는 확장성이 있음

6.1.2 SIARD 2.1 및 SIARD Suite 단점

(1) SQL:2008의 한계

- 현재 SIARD Suite은 SQL:2008 표준을 준수하고 있으며, 이 표준에 벗어나는 기능과 요소들을 수용하지 않는 구조임

- 하지만, 대부분의 RDBMS는 SQL:2008 표준에는 없는 기능과 요소들을 제공하고 있어서 각 RDBMS의 기능 및 요소 전체를 하나도 빠짐없이 SIARD 파일로 변환하기 어려움

- 특히, CREATE Table 구문의 경우 대부분 표준을 따르고 있지만, DBMS마다 SQL:2008에서 벗어난 특정 구문을 추가하는 형태를 가지고 있어서 SIARD 파일로 변환시 원래의 테이블과 완전 동일한 형태의 테이블을 구성하는 것이 어려움

- Stored Procedure/Function의 경우 DBMS별로 정의 방법이나 Stored Procedure/Function Body를 구성하는 방식이 달라서 SIARD 파일에도 간단하게 Stored Procedure/Function의 Attribute(Procedure/Function 이름) 정도만 저장하고 있음
(※ Oracle은 Procedure와 Function의 Body가 PL/SQL로 작성되어 있지만, 큐브리드에서는 JAVA Class로 되어 있음)

- 그래서 새로운 DBMS로의 확장을 하기 위해서는 JDBC Driver를 SIARD Suite에 단순히 붙였을 때 지원되는 요소들과 지원되지 않는 요소들로 구분하고 지원되지 않는 요소들에 대해서는 별도의 구현 과정이 필요함

(2) JDBC의 한계

- JDBC는 데이터베이스 인터페이스로 가장 많이 사용하는 인터페이스이며, 시스템 플랫폼이 Windows건 Linux건 상관없이 동작할 수 있다는 장점을 가지고 있어서 SIARD에서도 이를 적극 활용하였음

- 하지만, JDBC의 API 중에는 특정 요소 및 기능들과 관련된 API들이 DBMS로부터 모든 일부 데이터를 가지고 오지 못할 수 있음

- 예를 들어, Stored Procedure/Function의 선언 부분까지만 지원하고 있으며, Trigger 관련 API는 없음

- 또한, 큐브리드에서 제공하고 있는 SERIAL Type들에 대해서도 API가 없음

- 근본적인 이유는 앞서 언급된 SQL:2008의 한계와 마찬가지로 대부분 RDBMS 들이 각자 자신만의 요소 및 기능들을 제공하기 때문임

- 정리하면, JDBC는 DBMS 관련 다양하고 일관된 인터페이스로 폭넓은 사용성을 확보하

고 있지만 지원되지 않는 일부 기능으로 한계가 존재하므로, 전체 DBMS의 기능 및 요소들을 모두 SIARD 포맷으로 변환하기 위해서는 JDBC API 표준에서 벗어날 수 있는 별도의 구현이 필요함
(※ 본 프로젝트에서는 Trigger와 SERIAL 타입에 대한 JDBC API가 없어 추가적으로 구현함. 이는 JDBC API 범위에 벗어나는 것임)

(3) DBMS간 보존/복원의 한계

- SIARD에서 표방한 것 중 하나는 SIARD 파일을 통한 DBMS간 호환성, 즉 Oracle DBMS의 데이터베이스를 보존한 것을 MySQL이나 큐브리드로 복원하는 것임

- 그러나, 상호 호환성이 100% 보장되지 않는데, 이것은 SQL:2008 표준을 벗어난 데이터 타입도 문제 중 하나로 생각되지만 표준을 준수한 경우에도 문제가 됨

- DBMS간 데이터 전환 중 발생하는 데이터 타입 문제는 바로 자릿수(precision)로 DBMS마다 테이터 타입이 가지는 최대 자릿수가 다르기에 상호 데이터 전환에서 문제가 발생할 수 있음

- 예를 들면, String의 최대 허용 길이가 DBMS별로 다른데, 이 다른 길이의 차이로 인해 SIARD파일로 보관된 String 데이터가 복원되는 DBMS에 따라 일부가 손실되어 복원되거나 복원 자체가 불가할 수 있음
(단, SIARD파일에는 원래 String이 그래도 있음)

- 이외에도 Binary 데이터의 길이 또한 복원되는 DBMS에 따라 표현할 수 있는 최대 자릿수가 달라서 데이터 복원에 문제가 발생할 수 있으며, Stored Procedure/Function과 Trigger, 큐브리드의 SERIAL 기능도 DBMS마다 다르게 구현되어 있기 때문에 DBMS간 보존/복원에서 문제가 될 수 있음

(3) 단일 쓰레드(Single Thread) 구조의 한계

- 현재의 SIARD Suite은 JDBC 인터페이스로 DB 관련 모든 동작을 단일 쓰레드 연동하는 구조이며, 단일 쓰레드 구조에서는 멀티코어 프로세서를 장착한 어떤 하드웨어에서도 성능이 제한적일 수 밖에 없음

- 이를 개선하기 위해서는 멀티 쓰레드 구조로 변경을 해야 하며, 이 구조 변경은 SIARD의 전반에 걸쳐 코드 수정이 필요하기 때문에 단기간이 아닌 수개월 이상이 소요될 수 있는 개발 기간이 필요

- 만약 멀티 쓰레드로 구조를 변경한다면, DB의 멀티 연결 관리 및 병렬 처리를 고민해야 하며, SELECT 구문으로 모든 데이터를 가져오던 것을 SELECT ... WHERE Key-condition을 추가해야 함. 여기서 Key-Condition은 해당 테이블을 검색하기 위한 조건으로 병렬 처리의 핵심이 될 수 있음

(4) 단일 SIARD파일 생성

- SIARD는 DB에서 다운로드된 내용을 zip 형식의 파일로 압축하는 구조이며, 이 압축 파일은 단일 파일로 구성됨

- 하지만 DB의 크기에 따라 이 파일은 그 크기가 GB 단위로 커질 수 있으며, 이 단일 파일로 구성된 파일을 GUI 형태로 보기 위해서는 SiardGui JAVA Application을 사용하는데 그 성능이 상당히 떨어질 것으로 판단됨

- SIARD파일의 크기에 따른 성능 저하를 개선하려면 단일 파일이 아닌 여러 개의 파일로 분산하는 것도 하나의 방법이 될 수 있음

(5) 중단 및 재시작

- DBMS로부터 SIARD 파일을 Download하는 중에 DB서버가 다운되거나 네트워크 문제 혹은 그 외 다른 문제로 더이상 SiardGui 또는 SiardCmd가 수행될 수 없을 때, 변환을 완료하지 못한 SIARD 파일은 사용할 수 없으며, 이후 Download를 계속하려면 처음부터 다시 수행해야 함

- DB의 크기가 수백 GB 혹은 수 TB로 데이터가 대용량인 경우, SIARD파일 Download/Upload 작업 시간은 10일 이상 소요될 수도 있음.

- 이를 개선하기 위해서는 SIARD 파일 Download/Upload 중, 상태 정보를 꾸준히 별도의 기록으로 남겨야 하며, 이후 복구 시점에서 별도의 기록을 통해 중단된 시점부터 SIARD 파일 Download/Upload를 다시 할 수 있도록 해야 함

(6) DBMS 외부 저장 파일 미포함

- 대부분의 DBMS는 숫자, 문자, 날짜/시간 이외에 “파일”을 저장하기 위해서 BLOB Type을 제공함

- 그러나, 상당히 큰 비율로 국내 DB서버들에서는 파일 저장하기 위해서 BLOB Type을 사용하지 않고 문자열 타입으로 파일 시스템(예: C드라이브)에 저장되어 있는 파일 경로를 저장하고, 해당 경로에 파일을 저장하는 방식으로 구현되어 있음

- DBMS와 연계되어 있는 SW 또는 솔루션을 유지·관리 및 수정·보완할 때 효과적이기 때문에 대부분의 개발자들은 BLOB Type 대신 이러한 방식으로 구현하고 있음

- 이런 방식으로 구현되어 파일 시스템에 저장되어 있는 파일은 DBMS의 관리대상이 아니므로 SQL:2008에 포함되지 않으므로 SIARD 파일에도 패키징되지 않음

- 그러므로, 이를 위한 별도의 추가 수정·보완을 위한 개발이 필요하며, 이는 SIARD 표준 외 별도의 규격으로 추가되어야 함

(7) 전체 DB 변환 · 복원

- 현재 SIARD Suite은 해당 DBMS에 접속하는 사용자가 접근할 수 있는 모든 테이블을 변환하고 복원함. 데이터세트의 특정 테이블 또는 컬럼을 변환 · 복원할 수 없기 때문에 테이블 일부만 필요하더라도 전체 데이터세트를 모두 가져와야 함

- 특정 테이블 또는 컬럼을 변환·복원할 수 있는 별도의 개발이 필요하며, SIARD 표준도 확장되어야 함

6.2 SIARD의 데이터세트 보존포맷 활용을 위한 오픈소스 수정·보완 사항

- SIARD를 데이터세트 보존포맷으로 활용하기 위해서는 오픈소스인 SIARD Suite를 수정 · 보완해야 함
- 특히, 여러 국내외 DBMS 확장이 필요하며, 본 절에서는 새로운 DBMS 확장 지원을 위해 수정 · 보완해야 할 부분과 수행 과정을 제안함
- SIARD Suite에서 제공하는 Type과 해당 DBMS의 JDBC에서 제공하는 Type을 (1) 기본 Data Type, (2) 특수 Data Type, (3) Key Type, (4) Routine Type으로 분류
- 각 분류 별로 지원가능 여부를 조사 및 분석하여 "JDBC 단순 Type 매핑", "JDBC Type 매핑 및 별도 처리", "JDBC 새로운 Type 및 처리"로 구분하여 개발 및 구현을 진행함

6.2.1 DBMS마다 제공 기능과 요소의 차이로 인한 문제 및 수정/보완사항

○ SIARD를 보존포맷으로 활용하는 것은 RDBMS에서 저장되어 있는 데이터베이스를 보존하는 것으로, 특정 RDBMS로 정해져 있으며, 그 RDBMS는 현재 SIARD 지원 목록에 없다고 가정함

○ 본 절에서 설명하는 내용은, 큐브리드 확장 SIARD Suite을 개발하고 4개의 DBMS를 SIARD Suite으로 변환·복원 과정을 검증하면서 발생한 이슈와 상황을 기반으로 함

○ SIARD Suite의 지원 DBMS 목록에 없는 새로운 DBMS를 확장하려면 해당 JDBC Driver 추가하고, SIARD Suite에 맞는 JDBC Wrapper를 제작해야 함

※ 큐브리드의 경우, 큐브리드 JDBC Driver(cubrid_jdbc.jar)와 JdbcCubrid(jdbccubrid.jar)를 제작 및 추가함

○ 지원 목록에 포함하는 것을 넘어서, **해당 DBMS에서 제공하는 모든 기능을 SIARD 파일로 Download하고 다시 해당 DBMS로 Upload**하기 위해서는, 가장 먼저 SIARD Suite에서 미지원하는 Type들을 분석하고, SIARD Suite을 구성하는 여러 프로젝트들의 소스코드에 Type 지원을 위한 부가적인 기능을 구현하여야 함

(1) DBMS Type 분류

○ 대부분 DBMS에서 제공한 타입들은 (1) 기본 Data Type, (2) 특수 Data Type, (3) Key Type, (4) Routine Type으로 분류할 수 있음 (<표 98>은 큐브리드의 경우)

연번	구분	큐브리드 제공 Type
1	기본 Data Type	BIT, BIT VARYING, SHORT, INT, SMALLINT, BIGINT, NUMERIC, DECIMAL, FLOAT, REAL, DOUBLE, CHAR, VARCHAR, STRING, BLOB, CLOB, DATE, TIME, TIMESTAMP, DATETIME
2	특수 Data Type	SERIAL, Collection.SET, Collection.MULTISET, Collection.LIST, Collection.SEQUENCE, ENUM
3	Key Type	Primary Key, Foreign Key
4	Routine Type	Stored Procedure, Stored Function, Trigger

<표 98> 큐브리드 경우 Type 분류

(2) 요소 분류에 따라 SIARD Suite 구현 방향 설정

○ SIARD Suite에서 이들 요소들의 지원 가능한 요소들인지를 판단하기 위해서는 SIARD2.1 표준(SQL:2008)과 함께 SIARD Suite에서 DBMS 접속할 때 사용하는 표준 JDBC 인터페이스들의 상세 기능을 조사 및 분석해야 함

○ 실제 관련 사항들을 조사 및 분석한 결과, DBMS에서 제공하고 있는 다양한 요소들에 따라 다른 구현 방향이 존재함

○ SIARD Suite를 구성하고 있는 JDBC Driver와 Wrapper의 API들의 활용 및 수정 가능 정도에 따라 구현 방향을 결정하는 것이 가장 효율적일 것으로 판단됨

○ 그러나 실제 구현을 진행한 경험을 바탕으로, 기본 Data Type, 특수 Data Type, Key Type, Routine Type에 따라 구현의 방식과 난이도가 결정됨

○ <표 99>은 요소 분류별 큐브리드를 확장한 경험으로 예상한 구현 방향 및 난이도임

○ <표 99>에서 1, 2번은 대부분 SIARD2.1 표준의 범위에 포함될 수 있어 SIARD2.1과 호환되고 JDBC에서도 새로운 API를 만들지 않아도 되는 경우가 많음

연번	종류	기본 Data Type			
		확장 DBMS	SIARD 2.1	구현 난이도	구현 방향
1	기본 Data Type	정수, 실수, 문자, 문장, 이진형 등	SIARD 2.1 포함되어 있어 대부분 지원 가능	하	· JDBC 및 JDBC Wrapper에서 큐브리드의 기본 Data Type을 SIARD2.1의 해당 Data Type으로 단순 매핑
2	특수 Data Type	객체 형	SIARD 2.1에 포함되지 않으며 별도의 작업 필요	중	· 다른 Data Type으로 매핑 가능한 경우 JDBC 및 JDBC Wrapper에서 수정
				상	· 새로운 Data Type을 정의해야 하며 JDBC 및 JDBC Wrapper에서 수정
3	Key Type	PK, FK	SIARD 2.1 포함되어 있어 대부분 지원 가능	하	· 다른 DBMS JDBC Wrapper 단순 참고 및 매핑
4	Routine Type	Stored Procedure/ Function, Trigger 등	SIARD 2.1 포함되어 있지만 대부분 Body는 변환안됨	상	· 대부분 DBMS마다 다른 방식으로 구현되어 있고, 응용 프로그램이기 때문에 DBMS내에서 작업하여 해결되지 않고, 외부 환경(파일 시스템 등)과 연계하기 때문에 전문가 및 많은 노력이 필요

<표 99> 확장 DBMS에서 제공하는 Type 유형에 따른 구현 방향

○ <표 99>에서 2번의 난이도가 상인 경우는 SIARD2.1 표준의 범위에 포함되어 SIARD2.1과 호환되지만, JDBC에서 새로운 API를 만드는 경우 JDBC API 범위를 벗어나게 됨

○ <표 99>의 3번은 SIARD2.1 표준을 벗어나게 되어 생성된 결과물은 SIARD2.1과 호환되지 않음. 그러므로 SIARD2.1를 포함한 다른 표준(K-SIARD)로 재정의하여야 함

6.2.2 SIARD Suite 오픈소스 자체의 문제 및 수정/보완 사항

(1) 성능 및 안정성 개선

○ SIARD Suite SW 자체의 성능 문제 때문에 수정 및 보완이 필요

가. 단일 쓰레드(Single Thread) → 다중 쓰레드(Multiple Thread) 변경 필요

- 단일 쓰레드 구조에서는 멀티코어 프로세서를 장착한 어떤 하드웨어에서도 성능이 제한적일 수 밖에 없음

- 멀티 쓰레드 구조로 변경을 해야 하며, 이 구조 변경은 SIARD의 전반에 걸쳐 코드 수정이 필요하기 때문에 단기간이 아닌 수개월 이상이 소요될 수 있는 개발 기간이 필요

나. 단일 SIARD파일 구조 → 다중 SIARD파일 구조 변경 필요

- SIARD는 DB에서 다운로드된 내용을 zip 형식의 파일로 압축하는 구조이며, 이 압축 파일은 단일 파일로 구성됨

- DB의 크기에 따라 GB 단위로 커질 수 있으며, 이 단일 파일을 GUI로 확인하고 검색까지 하기 위해서는 성능이 상당히 떨어질 것으로 판단되며, 단일 파일이 아닌 여러 개의 파일로 분산하는 것도 하나의 방법으로 판단됨

다. 연결 재시작 수행을 위한 변경 필요

- DBMS로부터 SIARD 파일을 Download하는 중에 DB서버가 다운되거나 네트워크 문제 혹은 그 외 다른 문제로 더이상 SiardGui 또는 SiardCmd가 수행될 수 없을 때, 완료되지 못한 SIARD 파일은 사용할 수 없으며, 이후 Download를 계속하려면 처음부터 다시 수행해야 함

- DB의 크기가 수백 GB 혹은 수 TB로 데이터가 굉장히 큰 경우, SIARD 파일 Download/Upload 작업 시간은 10일 이상 소요될 수도 있음

- 이를 개선하기 위해서는 SIARD 파일 Download/Upload 중, 상태 정보를 꾸준히 별도의 기록으로 남겨야 하며, 이후 복구 시점에서 별도의 기록을 통해 중단된 시점부터 SIARD 파일 Download/Upload를 다시 할 수 있도록 해야 함

라. DBMS 외부 저장 파일을 위한 변경 필요

- 파일 저장하기 위해서 BLOB Type을 사용하지 않고 문자열 타입으로 파일 시스템(예: C 드라이브)에 저장되어 있는 파일 경로를 저장하고, 해당 경로에 파일을 저장하는 방식으로 구현되어 있는 경우, 파일 시스템에 저장되어 있는 파일은 DBMS의 관리대상이 아니고 SQL:2008에서는 이러한 경우를 고려하지 않기 때문에 SIARD 파일에 포함되지 않음

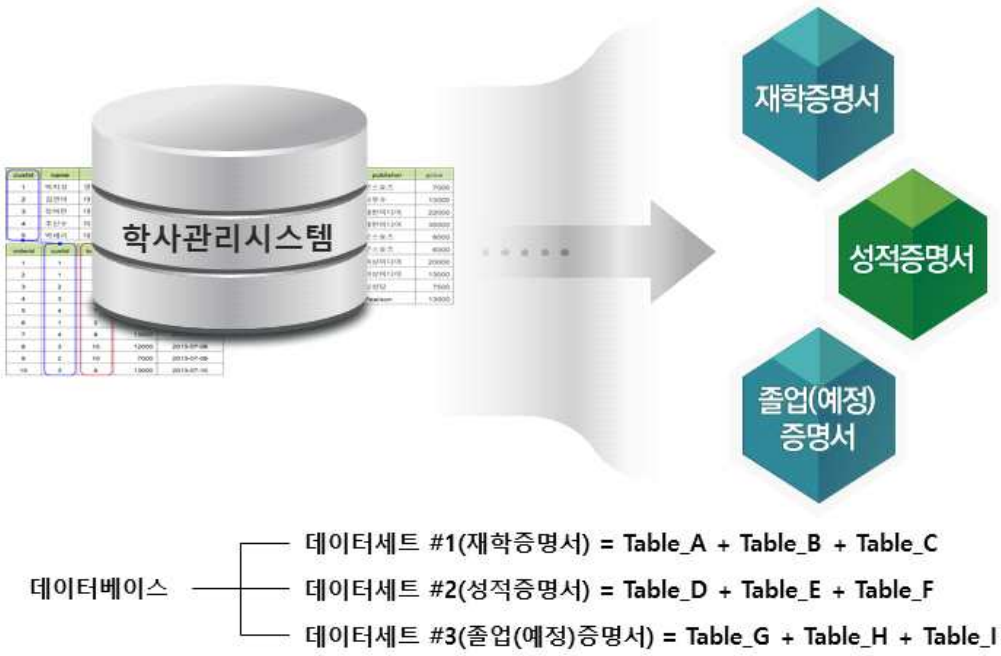
- 이러한 방식으로 파일 시스템에 저장되어 있는 DBMS 연관 파일들을 복원시키기 위해서는, 이를 위한 별도의 추가 수정·보완을 위한 개발이 필요하며, 이는 SIARD 표준 외 별도의 규격으로 추가되고 기존의 SIARD 파일과 논리적 또는 물리적인 연결을 시켜야 함

바. 전체 DB가 아닌 테이블 단위로 변환할 수 있는 기능 필요

- 하나의 물리적인 데이터베이스는 <그림 79>처럼 하나 이상의 논리적인 데이터세트로 구성되었다고 볼 수 있으므로 하나의 물리적인 데이터베이스 전체를 보존포맷으로 변환하

는 것보다 여러 개의 논리적인 데이터세트로 구분하여 보존포맷으로 변환하는 것도 고려되어야 함

- 이를 위해서는, 전체 데이터베이스를 논리적인 데이터세트로 구분하고, 각 데이터세트를 구성하는 테이블들을 분류할 수 있어야 함
- 특정 데이터세트를 구성하는 테이블들만 SIARD 파일로 변환 및 복원하기 위해서는 SIARD Suite에 테이블 단위로 가지고 오는 기능을 추가적으로 구현해야 함



<그림 79> 데이터 식별 예시
(학사관리시스템DB = 재학증명서 + 성적증명서 + 졸업(예정)증명서)

6.3 SIARD기반으로 데이터세트 보존 방안

- SIARD는 데이터세트 보존을 위해 제정되었고 이 표준을 준수하는 오픈소스(SIARD Suite)도 개발되어서, 데이터세트 보존포맷으로 가장 많이 논의되고 있음
- 이번 연구과제에서 SIARD를 조사·분석·개발·검증을 수행한 경험을 토대로, SIARD 표준이 DB형 데이터세트의 보조포맷으로 어떻게 적용될 수 있는지 검토하고 방안을 제시하고자 함
- SIARD 기반으로 데이터세트 보존하는 방향을 "장기보존" 관점(AIP: Archival Information Package)과 "활용" 의 관점(DIP: Dissemination Information Package)에서 적절성을 검토하고 장기보존 및 활용 방안을 제시함

6.3.1 데이터세트 보존 시 장기보존 관점에서의 SIARD 도입 적절성 검토

○ SIARD는 Software Independent Archival of Relational Databases의 약자로 "Relational Database(관계형 데이터베이스)"의 "Archival(장기보존)" 이 목적임

○ SIARD 표준 문서에서는 SIARD를 다음과 같이 관계형 데이터베이스의 장기보존을 위한 포맷이라고 설명하고 있음

*It is an open file format for **the long-term archiving of relational databases** in the form of text data based on XML that are packaged in a container file (SIARD archive)*

○ 또한, 아래의 제정 배경에서 알 수 있듯이 시간제한 없는 장기보존임을 명시하고 있으며, 사람이 읽을 수 있고 이해할 수 있는 방법으로 해석하고 보여 줄 수 있어야 함을 강조하고 있음

*Long-term archiving is the preservation, normally **without a time limit**, of the information stored in the SIARD files while retaining the bit stream and the ability to interpret and display the data in a way that is **human-readable** and **comprehensible**.*

○ 그래서, SIARD 표준은 Unicode, XML, SQL:2008, URI, ZIP 등의 공인받은 국제 표준을 기반으로 개발되었고, 데이터 자체를 Unicode로 인코딩하여 데이터와 함께 관계형 데이터베이스가 가지고 있는 테이블 구조까지 XML 형식으로 정의하고 보관하기 때문에 오랜 시간 후에도 데이터 값과 구조를 확인할 수 있음

○ 그러므로, 시간제한 없이 오랜 시간 후에도 확인할 수 있어야 한다는 장기보존의 가장 큰 목적에는 상당히 적합한 포맷이라고 할 수 있음

○ 데이터베이스에 저장된 데이터와 구조에 대한 장기보존에는 적합하지만, 데이터베이스가 다른 SW와 연동되어 사용자에게 데이터를 가공하여 보여 주는 서비스의 일부 구성요소로 사용되는 경우에는 데이터베이스와 함께 관련 SW를 함께 고려해야 하므로 다각적인 검토가 필요함

○ 데이터와 구조에 대해 SIARD를 보존포맷으로 활용하기 위해서는, SIARD Suite이 지원하는 DBMS일지라도 해당 DBMS만이 제공하는 특수한 데이터 및 요소를 보존할 수 있도록 수정이 필요함

○ SIARD Suite이 지원하지 않는 DBMS일 경우는 해당 DBMS의 데이터 및 요소를 지원하도록 해당 DBMS JDBC를 SIARD Suite의 구조에 맞게 추가 및 보완해야 함
('6.2 SIARD의 데이터세트 보존포맷 활용을 위한 오픈소스 수정·보완 사항' 참조)

6.3.2 데이터세트 보존 시 활용 관점에서의 SIARD 도입 적절성 검토

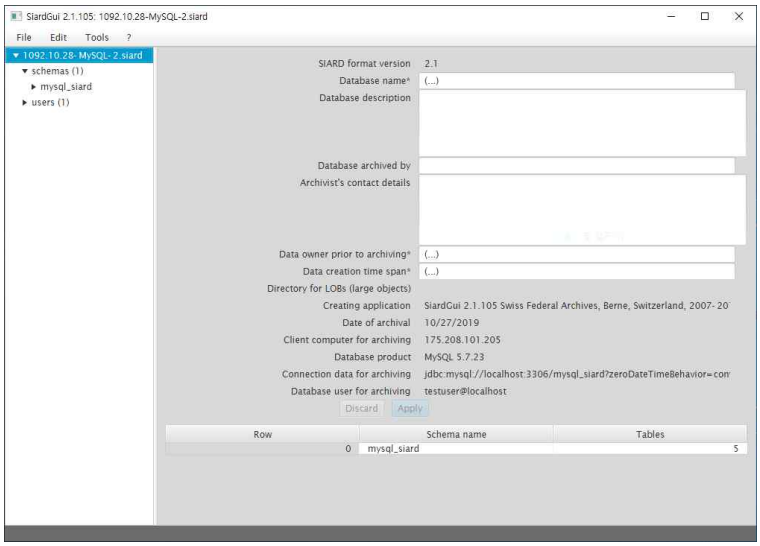
○ SIARD로 보존된 데이터세트를 활용하는 주 목적은 검색으로 빅데이터의 분석 및 AI의 학습자료로 활용될 수 있으며, 그 방법은 크게 2가지로 고려할 수 있음

○ 첫 번째 방법은 SIARD 파일 자체를 활용하는 방법이며, 두 번째 방법은 SIARD 파일을 DB에 복원시킨 다음 활용하는 방법임

6.3.2.1 SIARD 파일 자체 활용방안의 적절성 검토

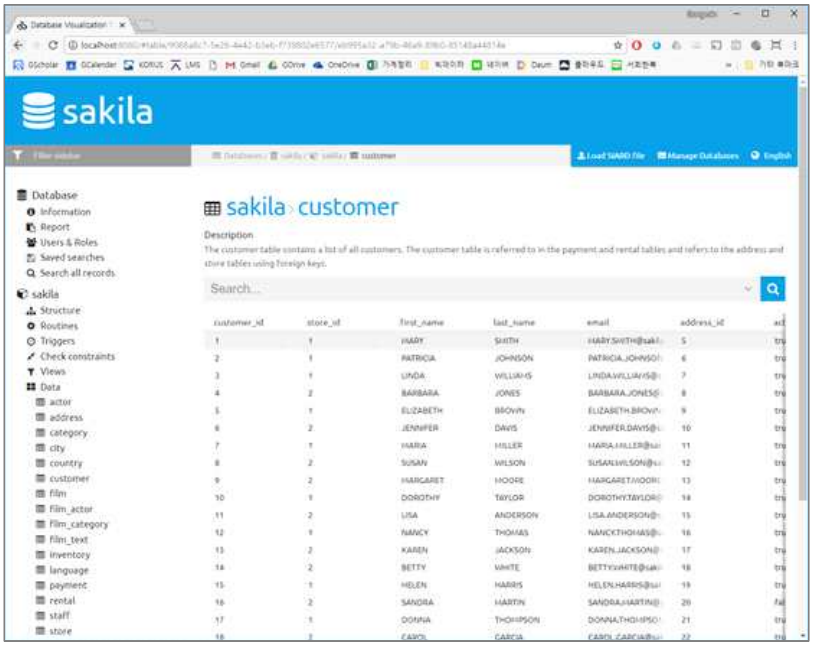
○ SIARD Suite에 의해서 생성된 SIARD 파일은 Unicode로 인코딩되어 있는 XML 구조의 텍스트 파일이기 때문에 메모장, 웹브라우저 등의 다양한 텍스트 뷰어 및 에디터로 확인할 수 있음

○ SIARD Suite의 SiardGui는 SIARD 파일의 데이터 및 구조를 보여 줄 수 있는 Viewer 기능(File → Open)을 <그림 80>처럼 제공하고 있음



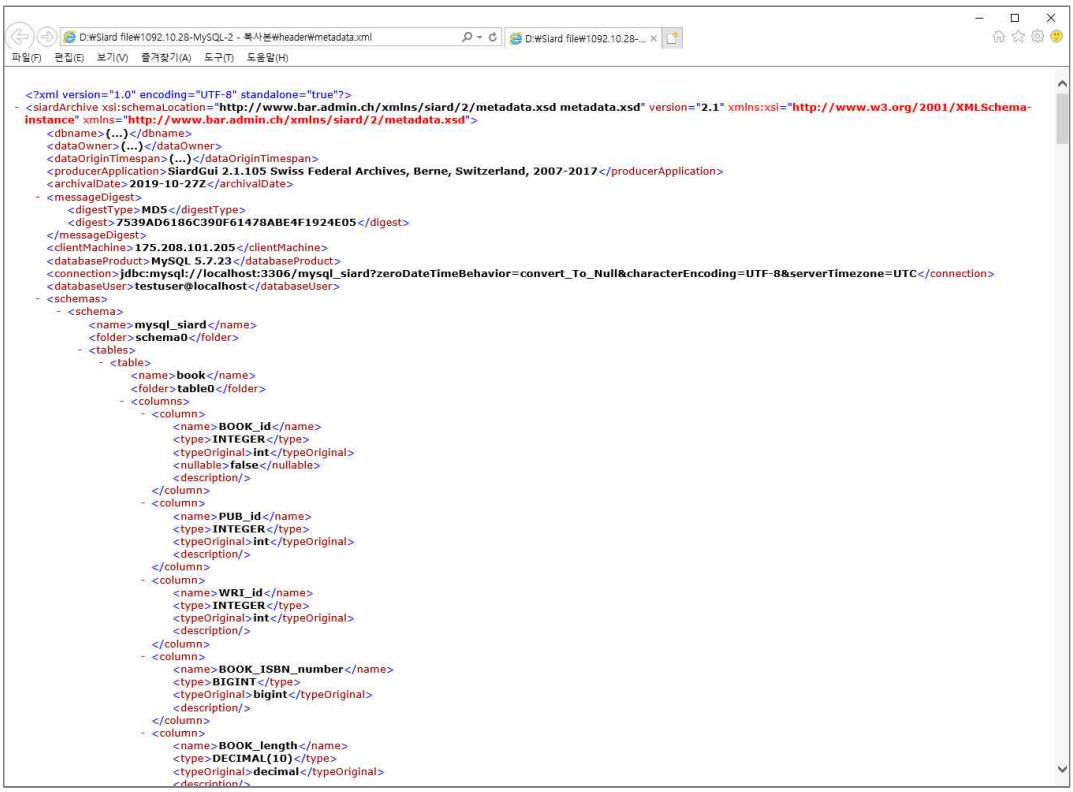
<그림 80> SiardGui의 SIARD 파일 Viewer 화면

○ 그리고, 포르투칼 기업인 KEEPS Solution에서 제작한 오픈소스 DBVTK(Database Visualization ToolKit)는 웹기반으로 제작되어 웹브라우저에서 SIARD 파일을 확인할 수 있는 Viewer임(<그림 81> 참고)

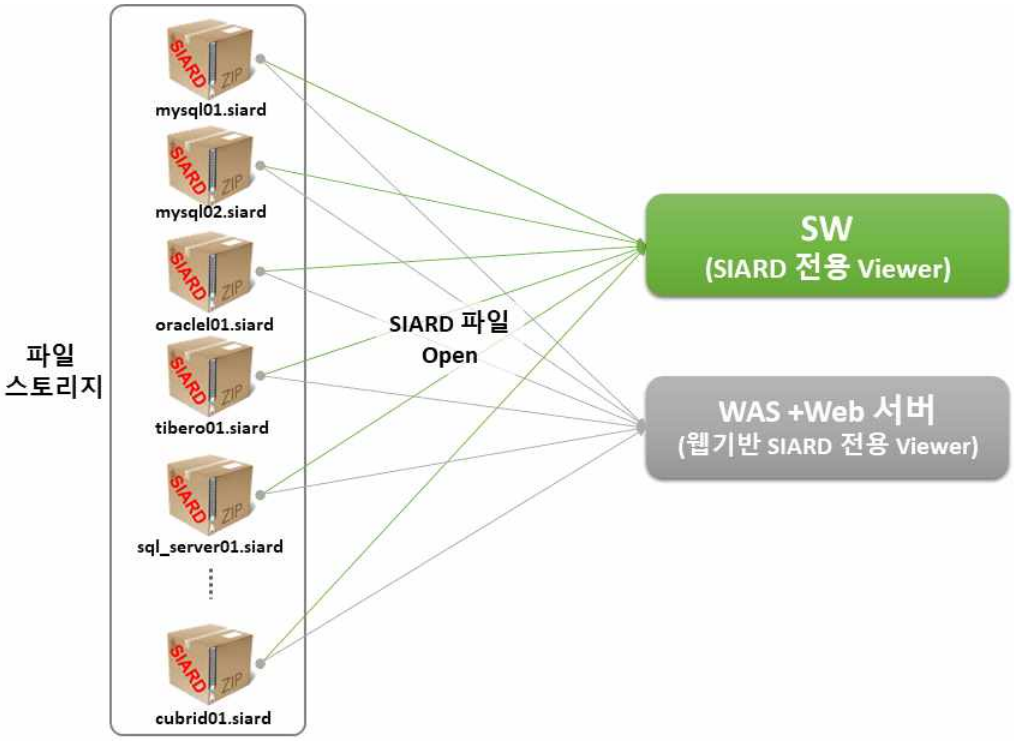


<그림 81> Database Visualization Toolkit SIARD 파일 Viewer 화면

○ 또한, SIARD 파일은 XML로 되어 있으므로, 웹브라우저에서도 <그림 82>처럼 쉽게 확인할 수 있음



<그림 82> Internet Explorer 웹브라우저로 SIARD 파일 확인 화면



<그림 83> SIARD파일의 자체 활용방안 예시

○ 그래서, SIARD 파일 자체 활용 방안은 <그림 83>처럼 SIARD 파일들을 대규모 파일 스토리지에 저장하고 있다가 SIARD 전용 Viewer 또는 웹기반 SIARD 파일 전용 Viewer로 SIARD 파일을 열어서 내용을 확인 또는 검색할 수 있음

○ 이때, 실제 활용하는 데에는 몇 가지 문제점들이 존재하며, <표 100>에 SIARD 파일 자체를 활용하는 데 있어 문제점과 해결방안을 정리하였음

연번	구분	내용
1	문제점	· SIARD Suite가 해당 DBMS의 모든 기능과 데이터를 변환하고, 다시 DB로 복원할 수 있도록 해야 함 ✓ SIARD Suite이 지원하지 않는 DBMS일 경우는, 해당 DBMS의 데이터 및 요소를 변환할 수 있도록 해당 DBMS의 JDBC를 SIARD Suite의 구조에 맞게 추가 및 보완해야 함 ✓ SIARD Suite이 지원하는 DBMS인 경우에도, 해당 DBMS만이 제공하는 특수한 데이터 및 요소를 보존할 수 있도록 수정이 필요
	해결방안	· '6.2 SIARD의 데이터세트 보존포맷 활용을 위한 오픈소스 수정·보완 사항' 참조

2	문제점	· DBMS의 가장 큰 목적은 검색을 위한 질의-응답 과정이며, 이 질의-응답은 SQL로 이루어짐 · SQL 질의-응답은 데이터가 DB에 있을 때만 가능하므로, 텍스트파일로 생성되는 SIARD 파일에서 SQL 질의-응답으로 검색할 수 없음
	해결방안	· 각 SIARD 파일이 DB에 존재했을 때 사용되었던 SQL 질의문을 별도로 확보하고, 그 SQL 질의문에 대해서 응답을 작성하는 SW를 별도로 제작해서 활용
3	문제점	· DBMS의 데이터는 일반적으로 대용량이기 때문에 이를 변환한 SIARD 파일도 대부분 대용량의 파일일 가능성이 큼 · 대부분 텍스트 또는 XML 파일의 내용을 보여 주는 응용 프로그램(메모장, 웹브라우저 등) 그리고 SIARD 파일 전용 Viewer(SiardGui, DBVTK 등)는 파일의 모든 텍스트 파일 내용을 주기억장치(메인 메모리)에 올려 놓은 후에 데이터를 보여 줌. 이때, 오랜 시간이 걸린 후에 열리거나 프로그램이 정지되는 문제를 확인함
	해결방안	· SIARD 파일 내용을 보여 주는 Viewer를 제작할 때, SIARD 파일 내용을 일부만 보여주고, 필요한 경우 추가적으로 파일 내용을 읽어서 가지고 올 수 있는 형태로 제작되어야 함 · 또한, 단일 파일이 아닌 여러 개의 파일로 분산하는 것도 하나의 방법일 수 있음

<표 100> SIARD 파일 자체 활용시 문제점과 해결방안

○ 이러한 문제들은 대규모 데이터세트의 경우 더욱 해결하기 어려울 것으로 판단되며, SQL 질의-응답이 제대로 지원되기 않으므로, 활용할 경우 사용자에게 큰 불편을 줄 수 있음

○ 또한, SIARD 파일 자체로 빅데이터 · AI에 활용되기 위해서는 SQL질의-응답으로 데이터를 추출하지 못하고 zip을 해제하여 XML 파일을 사용해야 하는데, SIARD 파일로 변환되기 전 DBMS에 있는 상태로 분석하는 것이 보다 효율적임

○ 데이터세트가 다른 SW와 연동되어 가공된 데이터를 제공하는 경우, SIARD 파일 자체로 활용하는 방안으로는 해당 기능을 재현하는 것은 불가능함

6.3.2.2 SIARD 파일을 DB로 복원하여 활용하는 방안의 적절성 검토

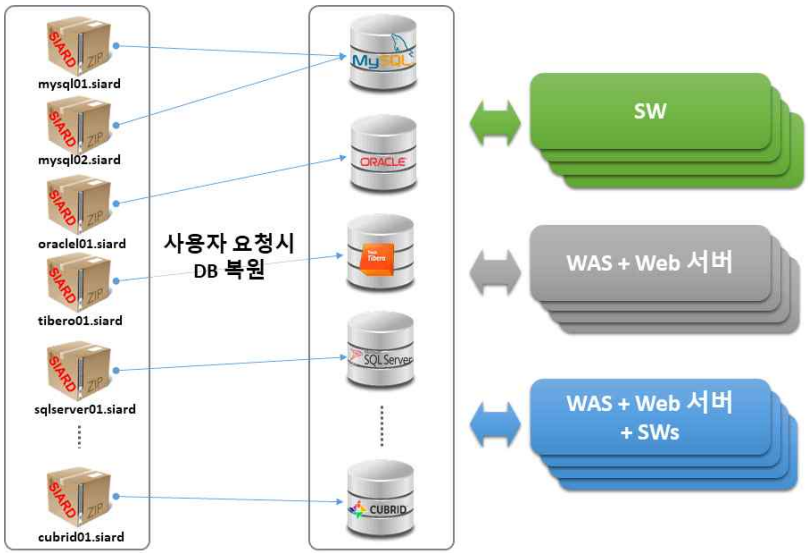
○ SIARD 표준은 DB로 데이터를 복원시키는 것을 목표로 제정되었으며, 서로 다른 DB 사이에서도 SIARD 파일 업로드가 가능하도록 SIARD Suite을 제작하였음

○ MySQL과 SQL Server 사이에서 한쪽의 DBMS으로부터 변환하여 SIARD 파일을 생성하고, 다른 한쪽의 DBMS에 SIARD 파일을 업로드하여 원본DB와 업로드DB 데이터를 비교하였더니 동일한 데이터임을 확인할 수 있었음(4.3.7-바 참고)

○ SIARD 파일이 DB에 복원된 후에는 SQL 질의-응답을 수행할 수 있기 때문에 SQL 수행이 가능한 SW가 접속하여 원하는 데이터를 검색할 수 있음(<그림 84> 참고)

○ 또한, DB복원이 되면 보존 전에 DB가 활용되기 위해 함께 연동되었던 WAS, Web서버, 다른 SW들이 구동된다면 예전 모습을 그대로 재현할 수도 있음(<그림 84> 참고)

○ SIARD 파일은 SQL질의-응답 기능을 수행할 수 있도록 DB복원이 가능한 포맷이므로 오랜 기간 동안 보존이 된 후에도 SIARD 파일만 보존되어 있다면 활용할 수 있다는 큰 장점을 가지고 있음



<그림 84> SIARD 파일의 DB복원 활용방안

○ 그러나, SIARD 파일로 변환되어 있는 다수의 데이터세트를 DB로 복원하여 원래의 서비스를 제공할 수 있도록 활용하기에는 비효율적 측면이 존재하며, 게다가 완벽하게 활용할 수 있는 환경을 구축하기까지 해결해야 할 문제점들이 존재함

○ 보존시기와 비슷한 시기에 분석 및 활용을 바로 고려한다면 SIARD 파일로 변환하고 활용을 위해서 복원하는 것은 비효율적임. 장기보존일 경우에만 SIARD로 변환하는 것이 바람직함

○ <표 101>에 보존되고 얼마 안 지나서 SIARD 파일로 변환되어 있는 다수의 데이터세트를 활용하는 데 있어 발생가능한 문제점과 해결방안을 정리함

○ 보존기간이 많이 지난 시기가 아니라 보존가 동시 또는 보존기간이 얼마 안 지나서 활용된다면 비효율적인 측면은 더욱 커질 것이며, 대규모의 시스템이 많을수록 DB가 다른 SW들과 연관도가 높을수록 더욱 해결하기 어려울 것으로 예상됨

연번	구분	내용
1	문제점	· SIARD Suite가 해당 DBMS의 모든 기능과 데이터를 변환하고, 다시 DB로 복원할 수 있도록 해야 함 ✓ SIARD Suite이 지원하지 않는 DBMS일 경우는, 해당 DBMS의 데이터 및 요소를 변환할 수 있도록 해당 DBMS의 JDBC를 SIARD Suite의 구조에 맞게 추가 및 보완해야 함 ✓ SIARD Suite이 지원하는 DBMS인 경우에도, 해당 DBMS만이 제공하는 특수한 데이터 및 요소를 보존할 수 있도록 수정이 필요 ✓ 단일 쓰레드(Single Thread)로 동작하기 때문에 성능이 저하되는 문제, 중간에 중단하면 처음부터 시작해야 하는 부분을 개선해야 함
	해결방안	· '6.2 SIARD의 데이터세트 보존포맷 활용을 위한 오픈소스 수정·보완 사항' 참조
2	문제점	· SIARD 파일의 DB복원 활용의 경우, SIARD파일을 저장하기 위한 파일 스토리지에 대한 비용 이외에 DBMS 시스템 및 라이선스, 이와 연동되어 있는 SW, WAS, Web 서버 등의 설치 및 라이선스 비용이 많이 소요될 것으로 예상됨
	해결방안	· 각 라이선스를 보유한 기업과 협의하여 국가적인 기록물 보존을 목적으로 한 별도의 라이선스를 확보할 수 있도록 협의하는 것이 필요함
3	문제점	· WAS, Web 서버 및 다양한 SW들이 연관되어 구동되었던 경우에는 서버를 구매하고 SW만 설치하더라도 원래의 서비스를 제대로 재현되지 않을 것으로 예상됨
	해결방안	· DB와 연계 동작하기 위해서 서버를 구매하고 SW만 설치하는 것 이외에 시스템 설정을 현재의 네트워크에 맞추고 동작이 제대로 되는지 검증해야 함 · 또한, 전자서명처럼 외부 네트워크에 인증이 필요했던 경우, SW를 그대로 설치하면 동작이 제대로 이루어지지 않기 때문에, 이 부분을 제외하거나 인증을 가상으로 받도록 수정 보완이 필요

<표 101> SIARD 파일을 DB로 복원하여 활용할 경우 발생가능한 문제점과 해결방안

○ 데이터세트가 다른 SW와 연동되어 가공된 데이터를 제공하는 경우에도 해당 기능을 재현할 수는 있겠지만, 이러한 시스템을 구축하는데 필요한 비용이 상당히 크며, 고도의 IT기술력을 갖춘 고급 기술자들 여러 명의 수개월 간의 노력이 필요한 작업으로 판단됨

○ DB복원하여 빅데이터 · AI에 활용함에 있어 SQL질의-응답이 가능하지만 SIARD 파일로 변환 및 DB복원하지 않고 DBMS에서 분석하는 것이 보다 효율적임

○ 또한, 이러한 시스템을 계속 유지하기 위한 유지 및 관리 비용도 높을 것으로 판단됨

6.3.3 데이터세트 보존 시 장기보존 및 활용 관점에서의 SIARD 도입 방안

구분	내용
장기보존	· SIARD로 데이터세트 보존하기 위해서는 다양한 DBMS들이 제공하는 데이터 및 요소를 지원하도록 해당 DBMS JDBC를 SIARD Suite의 구조에 맞게 추가 및 보완해야 함 · 그러나, SIARD 표준은 Unicode, XML, SQL:2008, URI, ZIP 등의 공인받은 국제 표준을 기반으로 개발되었고, 데이터를 Unicode로 인코딩하여, 관계형 데이터베이스가 가지고 있는 테이블 구조와 함께 XML로 보관하기 때문에 오랜 시간 후에도 데이터 값과 구조 확인가능 → 시간제한 없이 오랜 시간 후에도 확인할 수 있어야 한다는 장기보존의 가장 큰 목적에는 적합한 포맷이라고 할 수 있음
활용	· SIARD 파일 자체 활용:DBMS 완벽 지원을 위한 확장, SQL 질의-응답 기능 추가 등이 필요하며, 개발이 완료되더라도 원래의 DB가 다른 기능과 연계되어 제공했던 서비스 기능 재현이 불가능할 것으로 판단됨 · SIARD 파일의 DB복원 활용: 오랜 기간이 지난 후에도 SIARD 파일만 있으면 DB복원하여 기능 재현이 가능하지만, DBMS 완벽 지원을 위한 확장, DBMS 시스템 구축, DBMS연계 솔루션/시스템 구축 등이 필요하며, 이를 위한 예산은 시스템을 클라우드 환경으로 이관하는 규모와 비슷할 것으로 판단됨 → 대규모·대용량의 DBMS의 데이터세트를 SIARD파일로 변환하여 파일 자체를 활용하는 방안은 고려해야 할 요소 및 보완 사항이 많다고 판단됨 → 오랜 기간 동안 보존이 된 후에도 SIARD 파일만 보존되어 있다면, SIARD 파일은 SQL질의-응답 기능을 수행할 수 있도록 DB복원이 가능한 포맷이므로 SQL 질의-응답 기능을 수행할 수 있도록 활용할 수 있음 → 반면, 장기보존이 아닌 빅데이터 · AI 등을 통한 데이터세트 분석 및 활용 관점에서는 SIARD 변환 및 복원 과정은 불필요한 과정으로 판단됨

<표 102> 데이터세트 유형의 전자기록물에 대한 SIARD 도입 방안

제4장 제1세부연구개발과제의 연구결과 고찰 및 결론

1. 데이터세트 유형 전자기록 현황과 장기보존기술 조사

○ SIARD 2.1 표준은 메타데이터와 테이블데이터가 결합된 장기보존기술

- 원본의 데이터베이스 소프트웨어를 사용할 수 없게 되더라도 XML과 SQL:2008 표준에 기반하여, 데이터베이스 데이터의 접근과 교환을 가능케 할 수 있음

- 국외에서는 SIARD-DK 등 파생형이 활용되고 있으며, 세금계산서 등 실제 행정정보의 장기보존에 적용하는 연구가 진행 중

○ SIARD는 오픈소스 프로젝트이기 때문에 개발 및 배포 시 공개범위에 대한 검토 필요

- 행정정보데이터세트의 장기보존은 공공의 성격을 가지기 때문에, SIARD의 수정된 소스코드를 공개하는 것이 적합하다고 고려됨

2. 데이터세트 보존포맷 선정체계 수립 및 보존포맷 선정

○ 전자기록 보존포맷의 선정기준

- 파일포맷의 특성과 해당 기록유형의 특성을 근거로 보존포맷 선정기준 개발

· 보존포맷 선정을 위한 공통기준 : 파일포맷의 특성을 고려

· 보존포맷 선정을 위한 고유기준 : 기록유형의 특성을 고려

✓ 고유기준은 기록의 4대 요건을 충족시킬 수 있는 기준이 되어야 함

○ 모든 전자기록 유형에 적용되는 보존포맷 선정기준 : 공통기준

- 전자기록 보존포맷 선정기준의 현황과 파일 포맷의 특성을 근거로 도출

· 상위기준 : 5개

· 세부기준 : 10개

○ RDB형 데이터세트 보존포맷의 선정기준 : 고유기준

- SP(Significant Properties)를 적용하여 RDB형 데이터세트의 특성을 도출한 후 이를 근거로 고유기준을 도출

· RDB형 데이터세트 특성을 도출하는 데 적용할 수 있는 SP의 속성 : 3개(Structure, Content, Behavior)

· SP 속성을 근거로 RDB형 데이터세트 특성 도출

✓ RDB형 데이터세트 특성 : 5개(관계성, 다양성, 복잡성, 이질성, 상호작용성)

· RDB형 데이터세트의 특성을 근거로 도출

✓ RDB형 데이터세트 보존포맷 선정을 위한 고유기준 : 3개(일반화, 수용성, 활용성)

○ 보존포맷 선정 기준에 따라 평가 지표 개발

- 평가지표에 따라 권고 포맷 선정 : SIARD 검증

○ Non-DB형 데이터세트 보존포맷의 선정기준 : SP(Significant Properties)

- SP를 적용하여 Non-DB형 데이터세트의 특성을 도출한 후 이를 근거로 선정기준을 도출

· Non-DB형 데이터세트 특성을 도출하는 데 적용할 수 있는 SP의 속성 : 2개(Content, Structure)

· SP 속성을 근거로 Non-DB형 데이터세트 특성 도출

✓ Non-DB형 데이터세트 특성 : 3개(수용성, 활용성, 호환성)

○ 보존포맷 선정 기준에 따라 평가 지표 개발

- 평가지표에 따라 권고 포맷 선정 : BIFF8, CSV 검증

○ 연구결과의 고찰

- 다양한 유형의 전자기록에 대한 장기보존의 유연성 확보

· 전자기록 보존포맷 선정을 위한 기준을 마련하여 장기보존의 유연성 확보

- 다양한 유형의 전자기록에 대한 보존포맷의 확장성 확보

· 전자기록 보존포맷의 평가지표 개발을 통해 보존포맷 다양성 확보

- 다양한 유형의 전자기록 보존포맷 선정을 위한 고유기준 및 평가지표의 확장 필요

· 다양한 유형의 파일포맷 평가를 통해 평가지표를 지속적으로 보완

· 다양한 유형의 전자기록에 적합한 보존포맷 선정을 위한 고유기준 및 평가지표 보완

3. 국산 DBMS 대상 보존포맷 변환기능 개발

○ 변환기능 개발에 대한 고찰

- 단순히 JDBC를 확장만으로는 해당 DBMS의 모든 기능과 요소 전체 모두를 Download 및 Upload 할 수 없었음

- 확장하기 위한 DBMS의 기능과 요소들을 Data Type(기본/특수), Key Type, Routine

Type으로 구분하여 정리한 다음, SQL:2008 표준, JDBC API의 기능 명세를 함께 분석하면서 가능한 SQL:2008에 포함되도록 JDBC에 수정하거나 기능을 추가해야 함

- 만약 JDBC에서 수용할 수 없는 상황(큐브리드의 경우, Stored Procedure/Function)이라면 별도의 구현 과정이 필요함

4. 데이터세트 유형 전자기록 보존포맷 변환 검증

○ 보존포맷 변환 검증 시험 결과

- 2차례에 걸친 보존포맷 변환 검증 시험을 통해 SIARD 포맷 변환, 복구 검증 진행
- 검증 결과, SIARD는 Routine Type을 제외한 대부분 항목의 변환, 복구가 가능
- 하지만 Routine Type, MySQL의 “JSON”, Oracle의 “UROWID”와 같이 SIARD 변환이 지원되지 않는 항목과 안정성 확보를 위한 추가적인 개발이 필요

○ 보존포맷 변환 검증 시험 결과 고찰

- 내부 검증 4종의 DBMS(MySQL, SQL Server, Oracle 큐브리드)를 대상으로 진행한 검증 시험 결과, Data Type, Data, PK, FK 등 관계형 데이터세트의 보존에 중요한 정보 및 항목들을 SIARD 포맷으로 안정적인 변환 및 보존이 가능함

※ Oracle의 PK, FK 경우는 SIARD 포맷의 문제가 아닌 Oracle DBMS의 구조적인 문제로 판단

- 서로 다른 DBMS로 Download 및 Upload를 진행해도 SIARD 파일 내의 Data가 정상적으로 변환, 보존이 되기 때문에 보존포맷으로서 SIARD 포맷의 기능은 출중하다고 판단
- 하지만 대규모 DB를 SIARD 포맷으로 변환할 경우, 많은 시간이 소요되고 변환 도중 문제가 발생하면 처음부터 변환 작업을 해야한다는 단점이 존재함
- SIARD 포맷은 구조적으로 안정성을 확보할 수 있는 기능을 추가적으로 개발하는 것이 필요하다고 판단
- 또한, 검증 시험을 통해 최소 7GB, 180만 개 규모를 가진 DB는 안정적으로 SIARD 포맷으로 변환이 가능하다고 판단함

5. 실데이터에 대한 보존포맷 변환 검증

○ 큐브리드 DBMS 대상 실데이터 검증 결과 고찰

- 큐브리드에 대한 모든 기능과 요소들이 Download 및 Upload 될 수 있도록 SW를 개발하였기 때문에 DBMS 대상 검증 시험 결과는 원본DB와 복원DB 모두 동일하다고 나왔음

○ Oracle DBMS 대상 실데이터 검증 결과 고찰

- 결과는 모두 동일하다고 나왔지만, DATE Type의 데이터가 일부 손실되어 추가적인 SW 개발이 진행되었음

- Oracle DBMS에서 제공하는 기능 및 요소가 다른 모든 DBMS의 합집합 모두 크기 때문에 이 Oracle DBMS의 모든 기능과 요소를 Download 및 Upload 할 수 있으면 다른 DBMS로 확장할 때 많은 도움이 될 것으로 판단됨

6. 테스트베드 구축 · 시험 결과에 따른 장기보존방안

○ SIARD를 데이터세트 보존포맷으로 활용하기 위해서는 오픈소스인 SIARD Suite를 수정 보완해야 함

○ 특히, 다양한 DBMS 지원이 필요하며, 본 절에서는 새로운 DBMS 지원을 위해 수정해야 할 부분과수정 및 보완하는 과정을 설명함

○ SIARD Suite에서 제공하는 Type과 해당 DBMS의 JDBC에서 제공하는 Type을 (1) 기본 Data Type, (2) 특수 Data Type, (3) Key Type, (4) Routine Type으로 분류

○ 각 분류 별로 지원가능 여부를 조사 및 분석하여 “JDBC 단순 Type 매핑”, “JDBC Type 매핑 및 별도 처리”, “JDBC 새로운 Type 및 처리”인지를 구분하여 구현을 진행함

제5장 제1세부연구개발과제의 연구 성과

1 활용성과

세부과제명	데이터세트 유형 전자기록 보존포맷 선정 및 테스트베드 구축・시험・검증
세부과제책임자	양동민 / 전북대학교 기록관리학과 부교수, 문화융복합아카이빙 연구소 공동연구원 / 컴퓨터공학(컴퓨터네트워크・기록정보보안)

가. 연구논문

번호	논문제목	저자명	저널명	집(권)	페이지	Impact factor	국내/국외	SCI여부
1	클라우드 컴퓨팅 기반 에뮬레이션 전략을 활용한 전자기록 장기보존 방안 연구	이봉환, **한희정**, **조철용**, **왕호성**, **양동민**	한국기록관리학회지	19권 4호	1-33	-	국내 (KCI)	X

나. 학술발표

번호	발표제목	발표형태	발표자	학회명	연월일	발표지	국내/국제
1	A Study on the Long-Term Preservation of Digital Information Resources	Poster	**한희정** **양동민**	2019 ICLIS (International Conference on Library and Information Science)	2019.07.12. ~ 2019.07.13	Taipei, National Taiwan Normal University	국제
2	A Study on Administrative Information Datasets as Evidence of Public Service		**이정은** 윤은하 김건		2019.07.12. ~ 2019.07.13		국제

다. 지적재산권

번호	출원/등록	특허명	출원(등록)인	출원(등록)국	출원(등록)번호	IPC분류

라. 정책활용

· 공공기관의 데이터세트 유형 전자기록 보존 및 활용을 위한 지침 수립 지원
· 전자기록물의 마이그레이션 전략에서의 문서보존포맷 선정 체계 및 보존 방안

마. 타연구/차기연구에 활용

· 국가기록원 데이터세트 유형의 마이그레이션 전략을 위한 기술규격 표준화 연구 및 개발에 활용

바. 언론홍보 및 대국민교육

· 특기사항 없음

사. 기타

· 특기사항 없음

제6장 기타 중요변경사항

(참여인력 변경)

○ 큐브리드 확장 SIARD Suite 개발팀 소프트웨어 개발자를 큐브리드 DBMS 전문가로 변경

제7장 참고문헌

【국내】

국가기록원 (2008), 전자기록물 문서보존포맷 기술규격(NAK 30:2008(v1.0)), 2008.

국가기록원 (2007). 행정정보시스템 데이터세트 기록관리 방안 연구보고서.

국가기록원 (2015). 데이터세트 구조분석 및 진본성 보장 기록관리 기능모델 연구.

박병주 (2011). 데이터베이스 영구보존을 위한 디지털 아카이빙 보존포맷 및 도구 개발. 충남대학교 석사학위논문.

오세라, 박승훈, 임진희 (2018). 행정정보데이터세트 사례 조사 연구. 한국기록학회지. 109-133.

왕호성, 설문원 (2017). 행정정보데이터세트 기록의 관리방안. 한국기록관리학회지. 17(3). 23-47.

행정안전부, 한국정보화진흥원 (2018). 2018년도 범정부EA기반 공공부문 정보자원 현황 통계 보고서.

【국외】

Abrams, Stephen et al, (2005). "PDF-A: The Development of a Digital Preservation Standard.", Paper presented at the 69th Annual Meeting for the Society of American Archivists, New Orleans, Louisiana, August 14 - 21.

Adams (2008). "The National Archives. Digital Preservation Guidance Note: Selecting File Formats for Long-Term Preservation.".

Barnes, Ian. (2006). "Preservation of Word Processing Documents." . *Australian Partnership for sustainable repositories*.

CENDI Digital Preservation Task Group. (2007). "Formats for Digital Preservation: A Review of Alternatives and Issues".

Clausen, Lars R. (2004). "Handling File Formats". *Statsbiblioteket*.

Eun G.Park & Sam Oh. (2012). "Examining Attributes of Open Standard File Formats for Long-term Preservation and Open Access.", *Information Technology and Libraries.* 31(4), 46-67.

Folk, Mike, and Bruce Barkstrom. (2003). "Attributes of File Formats for Long-Term Preservation of Scientific and Engineering Data in Digital Libraries.", Paper

presented at the Joint Conference on Digital Libraries, Houston, TX, May 27 - 31.

ECMA. (2008). "Office Open XML File Formats－Part 1." 2nd ed.

Hodge, Gail and Nikkia Anderson. (2007). "Formats for Digital Preservation: A Review of Alternatives and Issues.", *Information Services & Use* 27: 45 - 63.

InterPARES. (2006). General Study 11 Final Report: Selecting Digital File Formats for Long-Term Preservation, InterPARES 2 Project.

Johnson, Amy Helen. (1999). "XML Xtends its Reach: XML Finds Favor in Many IT Shops, but It's Still Not Right for Everyone.", *Computerworld*, 33(42): 76 - 81.

Knight, Gareth. (2008). Framework for the definition of significant properties. The National Archives, InSPECT Project Document.

Lesk, Michael E. (1995). "Preserving Digital Objects: Recurrent Needs and Challenges.", *In Proceedings of the 2nd NPO Conference on Multimedia Preservation.* Brisbane, Australia.

Lindley, andrew. (2013). "Database Preservation Evaluation Report - SIARD vs. CHRONOS - Preserving complex structures as databases through a record centric approach?". *Conference: International Conference on Preservation of Digital Objects (iPres)*, At Lisbon.

Malcolm Todd. (2009). "File formats for preservation.", DPC Technology Watch Series Report 09-02.

Markus Hamm, Christoph Becker. (2011). "Report on decision foctors and their influence on planning.", Scalable Preservation Environment.

Mette van Essen, Maurice de Rooij, Bill Roberts, Maurice van den Dobbelsteen (2011). Database Preservation Case Study: Review. National Archives of the Netherlands.

Müller, Eva et al. (2003). "Using XML for Long-Term Preservation: Experiences from the DiVA Project.", In Proceedings of the Sixth International Symposium on Electronic Theses and Dissertations. Berlin, May: 109 - 116.

Puglia, Steven, Jeffrey Reed, and Erin Rhodes. (2004). "Technical Guidelines for Digitizing Archival Materials for Electronic Access: Creation of Production Master Files－Raster Images.", US National Archives and Records Administration.

Rog, Judith, and Caroline van Wijk. (2008). "Evaluating File Formats for Long-term Preservation.", National Library of the Netherlands.

Sullivan, Susan J. (2006). “An Archival/Records Management Perspective on PDF/A.”, Records Management Journal 16(1): 51 - 56.

TNA(The National Archives). (2008). Selecting File Formats for Long-Term Preservation.

Ricardo Andre Pereira Freitas. (2011). Siginificant Properties in the Preservation of Relational Database.

Wilson, Andrew. (2007). Significant Properties Report. InSPECT : Significant Properties Report.

van Wijk, Caroline, and Judith Rog. (2007). “Evaluating File Formats for Long-Term Preservation.”, Presentation at International Conference on Digital Preservation, Beijing, China, Oct 11 - 12, 2007.

【인터넷자료】

The National Arvhives(TNA), <http://www.significantproperties.org.uk/>, 2019.08.03., 확인.

Arms, Caroline R. and Carl Fleischhauer. , “Sustainability of Digital Formats: Planning for Library of Congress Collections.”, <https://www.loc.gov/preservation/digital/formats/index.shtml>, 2019.08.09., 확인.

Frey, Franziska, “5. File Formats for Digital Masters.”, In Guides to Quality in Visual Resource Imaging, Research Libraries Group and Digital Library Federation., <https://pdfs.semanticscholar.org/d63d/d3c6515fafb5fdf4925a26cac2e799436a0d.pdf>, 2019.08.09., 확인.

LAC(Library and Archives Canada), Guidelines on File Formats for Transferring Information Resources of Enduring Value, <http://www.bac-lac.gc.ca/eng/services/government-information-resources/guidelines/Pages/guidelines-file-formats-transferring-information-resources-enduring-value.aspx>, 2019.08.09., 확인.

LOC(Library of Congress), Digital Preservation at the Library of Congress - Sustainability of Digital Formats: Planning for Library of Congress Collections, <http://www.loc.gov/preservation/digital/formats/index.html>, 2019.08.09., 확인.

MSA(Minnesota State Archives, Minnesota Historical Society), Electronic Records Management Guidelines File Formats, <http://www.mnhs.org/preserve/records/electronicrecords/erfformats.php>,

2019.08.09., 확인.

NARA(National Archives), Federal Records Management - Frequently asked questions about Selecting Sustainable Formats for Electronic Records, <http://www.archives.gov/records-mgmt/initiatives/sustainable-faq.html>, 2019.08.09., 확인.

WHS(Wisconsin Historical Society), Best Practices for the Selection of Electronic File Formats, <https://www.wisconsinhistory.org/Records/Article/CS15427>, 2019.08.09. 확인.

제8장 첨부 및 별첨 목록

가. 첨부 서류 목록

[첨부01] 2019 ICLIS 대만학회 발표자료 1

[첨부02] 2019 ICLIS 대만학회 발표자료 2

[첨부03] 한국기록관리학 회 논문지 19권 4호 2019년 11월 게재(KCI)

[첨부04] 용어정리

나. 별첨 서류 목록

[별첨01] SIARD 2.1 표준 번역본

[별첨02] SIARD 2.1 표준 규격 분석 자료

[별첨03] SIARD Suite 빌드 및 SiardGui 실행 방법

세부 연구과제 요약

과제 고유번호	자동부여			공개가능여부	공개
주관과제명	데이터세트 유형 전자기록의 장기보존기술 연구				
제 1 세부과제명	데이터세트 유형 전자기록 보존포맷 선정 및 테스트베드 구축・시험・검증				
연구책임자	성 명	양 동 민			
	소속 기관명	전북대학교 기록관리학과, 문화융복합아카이빙 연구소 공동연구원			
	전자우편	*****	전화번호	***-****-****	

○ 연구목표 (400～600자)

· 데이터세트유형 전자기록현황 및 장기보존기술 조사
· 데이터세트 유형 전자기록 보존포맷 선정기준 수립 및 선정
· DBMS 데이터세트 유형 전자기록 보존포맷 변환 검증 시험 및 큐브리드 변환 · 복원 기능 개발

○ 연구내용 (1000～1200자)

· 데이터세트 유형 전자기록 현황 및 장기보존기술 조사 및 분석
 * 공공기관의 행정정보시스템에서 생산・관리되는 데이터세트 형태・운영・관리 현황
· 데이터세트 유형 전자기록 보존포맷 선정기준·평가체계 수립 및 보존포맷 선정
 * 데이터세트 유형 전자기록의 보존포맷 선정기준 수립 및 문서보존포맷 제시
· DB형 데이터세트 유형 전자기록 보존포맷 변환 · 복원 검증 시험
 * 큐브리드 대상 문서보존포맷 변환 검증 테스트베드 구축 및 검증 시험
· 큐브리드 변환 · 복원 기능 개발
 * 큐브리드를 대상으로 보존포맷 변환 · 복원 기능 개발
· 테스트베드 구축·시험 결과에 따른 보존적합방식 제안
 * RDB의 경우, 시스템・SW 환경, 시스템・SW 연계상황 등에 따라 보존방안 제안
 * 공공기관의 데이터세트 유형 전자기록 보존 및 활용을 위한 지침 작성 및 제출
 * 데이터세트 유형, 시스템 구성, 운영 형태 등에 따라 보존 및 활용 방안 제안
 * 마이그레이션 및 에뮬레이션 적용 시, 변환, 보존, 검증, 활용을 위한 지침 작성

○ 연구성과(응용분야 및 활용범위포함) (400~600자)

· 응용 분야 · 공공 전자기록물 저장 및 관리 서비스 · 민간 기업 데이터 저장 및 관리 서비스 · 데이터 비즈니스 마케팅 산업의 데이터 저장 관리 서비스 · 활용 범위 · 공공기관의 공공 데이터, 전자기록물 보관 및 관리 · 민간기업의 사원, 계약 정보 등 비즈니스 데이터 관리 · 데이터 비즈니스 기업의 생산, 재생산 데이터 관리 · 다양한 환경에서의 전자기록물 데이터 포맷 및 애플리케이션 테스트

○ 참여연구원

성 명	소속/직위	성 명	소속/직위
양동민	전북대학교 기록관리학과/부교수, 문화융복합아카이빙연구소/공동연구원	윤성호	전북대학교 기록관리학과/석사과정
한희정	전북대학교 문화융복합아카이빙연구소/전임연구원	조현우	전북대학교 기록관리학과/석사과정
박태연	전북대학교 문화융복합아카이빙연구소/전임연구원	소정의	전북대학교 기록관리학과/석사과정
이정은	전북대학교 기록관리학과/박사과정	김병욱	큐브리드/전문위원

Keywords (5개 내외)	*한글*	전자기록물, 데이터세트, 장기보존기술, 마이그레이션
	영문	electric records, dataset, long-term preservation technology, migration

제2세부연구개발과제 연구결과

클라우드 기반 전자기록의 장기보존기술개발 테스트베드 구축 및 에뮬레이션 시험·검증

조 철 용

㈜이노그리드

제1장 제2세부연구개발과제의 연구개발 목표

1. 제2세부연구개발과제의 목표

1.1. 연구배경 및 목적

○ (O/S 및 애플리케이션의 변화) 기술의 발달과 함께 문서의 저장 포맷 및 유형이 지속적으로 변화하고 있어 전자기록물의 호환성 문제 발생

- 보석글, 훈민정음, 하나워드부터 등 스프레드시트 유형의 프로그램들과 웹에디터 등 더 이상 사용하지 않는 다양한 애플리케이션 프로그램들이 존재

- 현재 가장 많이 사용하는 한글, 엑셀, 파워포인트 등 애플리케이션도 다양한 버전이 존재하여 버전 간의 파일 포맷 및 속성의 호환에 문제가 발생되고 있음

- 전자기록물 원본 형식을 지원하는 애플리케이션을 보관하고 있지만 운영환경이 변화하거나 컴퓨팅 환경이 변화할 경우 보관된 애플리케이션을 더 이상 수행할 수 없는 상황이 발생

○ (디지털 컴포넌트 기반의 전자기록물 재현 기술 한계) 최근 컴퓨팅 환경이 클라우드, 모바일 등으로의 변화로 현재의 전자기록물 재현 기술로는 빠르게 변화하고 있는 컴퓨팅 환경을 수용하기 어려움

- 현재까지 개발된 대부분의 애플리케이션은 MS의 윈도우 운영환경에서 동작하도록 개발되어 있으나, 운영 환경이 클라우드 서비스 환경으로 바뀔 경우 서버 환경에서 구동되어야 함

- 하지만 이미 사양화된 애플리케이션을 클라우드 환경에서 동작하도록 변환하는 것은 기술적·경제적 측면에서 어려움

- 더욱이 모바일 환경이 급속히 확산되고 클라우드 컴퓨팅 시대가 눈앞에 다가오는 시점에서 현재 개인용 컴퓨터의 90% 이상을 차지하는 윈도우 운영체제 구조에서 브라우저 기반의 운영체제로 바뀔 경우에는 현재 보관된 모든 애플리케이션을 실행할 수 있는 기반이 사라지므로 이러한 문제를 해결하기 위한 방안 연구가 필요

○ (에뮬레이션 연구에 대한 관련 분야의 요구 증대) 클라우드, 모바일 등 컴퓨팅 환경의 변화를 고려하여 동적 요소를 포함한 전자문서의 영구보존을 위한 에뮬레이션에 대한 깊이 있는 연구가 필요

- 대표적인 해외의 에뮬레이션 프로젝트인 KEEP, Planets, CAMiLEON, BK 등은 전자문서류 뿐만 아니라 다양한 OS와 미디어 응용 프로그램, 게임, 웹페이지 등을 재현 대상으로 선정하여 에뮬레이터를 개발하고 있음

- ※ EU의 KEEP 프로젝트는 디지털 객체의 장기보존에 대한 대표적 표준인 ISO 14721(OAIS)의 DIP 단계에서 렌더링하여 사용자에게 에뮬레이션 결과를 보여주는 개념을 제시

- DIP를 에뮬레이션하는 방식은 이미 패키징된 원본 AIP를 교체하지 않아 디스크 보존 용량을 변화시키지 않으며, 다른 장기보존 솔루션에 영향을 주지 않음

- 국내 공공기관에서 생산 및 접수하고 있는 기록물의 형태 중 동적 요소를 포함하고 있는 장기보존 대상을 평가하고 기술적 구현이 가능한 수준에서 에뮬레이터를 개발하는 것이 바람직함

- 컴퓨팅 기술이 발전하면서 시각과 청각에 의존하는 동적 요소에 대한 기술 개발이 가속화

- 따라서 기록 보존의 관점이 아니라 사용자의 기능 편의성을 제공하는 관에서 디지털 컴포넌트를 생성하는 애플리케이션의 기능이 보다 다양하고 복잡한 동적 렌더링 객체 요소를 포함하게 될 것으로 예상

○ (클라우드 기반 전자기록 관리 요구사항 증대) 데이터세트 유형의 전자기록 관리를 위해 에뮬레이션 전략으로의 확장 필요

- 최고 수준의 무결성, 진본성을 보장하기 위한 테스트베드 기반 연구가 필요

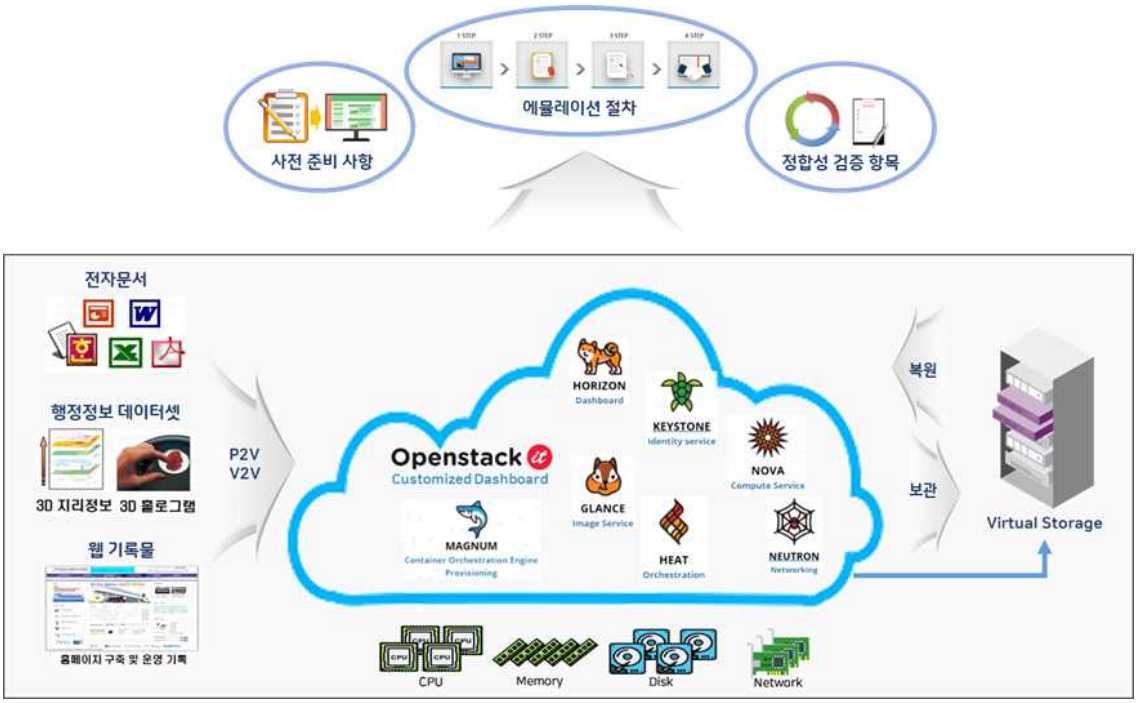
- 에뮬레이션 방식은 생상된 당시의 형태와 기능 그대로를 재현할 수 있는 가장 좋은 방법이지만 고도의 IT기술이 요구되고 비용도 많이 소요된다는 단점을 지님

- 최근 클라우드 기반 가상화 기술의 발달로 비용이 많이 절감되어 산업계에서는 이미 상용화가 이뤄져 널리 사용되고 있으며 Amazon 의 AWS(Amazon Web Service)가 대표적임

- DBMS의 데이터세트는 독자적으로 사용되지 않고 WAS/WEB서버들과 연계되어 운용되는 경우가 많기 때문에 데이터세트만 보존하기보다 시스템 전체를 보관하기 위한 에뮬레이션 방식이 적합한 경우가 많음

- 클라우드, 모바일 등 환경의 변화를 고려하여 동적 요소를 포함한 전자문서의 영구보존 및 에뮬레이션 실증을 위해 테스트베드 구축과 함께 데이터세트의 유형, 규모, 환경 등에 따른 실험과 검토가 필요

1.2. 연구의 목표



<그림 85> 클라우드 기반 에뮬레이션 시험 구성도

○ 보존방식에 따른 기술적합도 검증 테스트베드 구축

- 에뮬레이션 시험 검증을 위한 하드웨어 시스템 구축
- 오픈스택 기반 클라우드 인프라 환경 구축

○ 데이터세트 유형 전자기록의 에뮬레이션 시험

- 선정된 데이터세트의 에뮬레이션 후 원천 데이터세트와의 정합성 검증 항목 마련 및 점검
- 에뮬레이션 사전 준비, 에뮬레이션 절차, 정합성 검증항목 등 도출

2. 제2세부연구개발과제의 목표달성도

○ 본 연구팀은 당초 계획했던 연구 목표를 모두 달성하였음

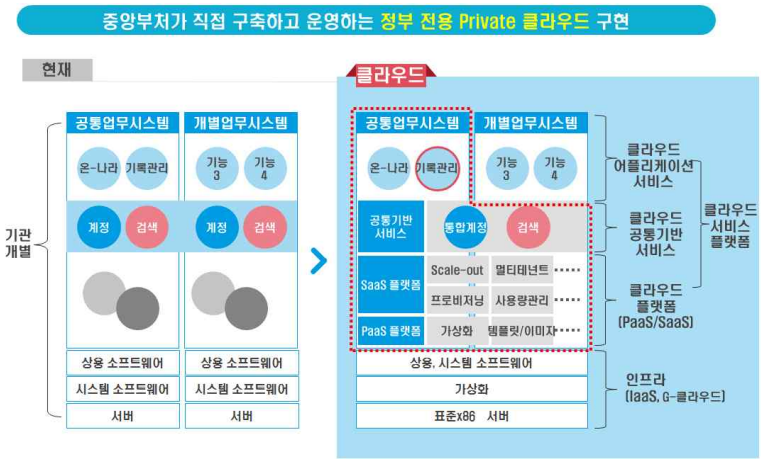
연구개발 추진내용			연구개발 일정							달성도
			5	6	7	8	9	10	11	
제2세부과제	에뮬레이션	**기술적합도 검증을 위한 테스트베드 구축**								100%
		.하드웨어 시스템 도입 및 구성								100%
		.클라우드 환경 구축								100%
		. 에뮬레이션 시험 환경 구축								100%
		데이터세트 유형 전자기록의 에뮬레이션 시험								100%
		.에뮬레이션 시험 대상 시스템 선정								100%
		.선정된 시스템 별 에뮬레이션 시험 검증								100%
		.에뮬레이션 절차, 정합성 검증 항목 도출								100%
		.전자기록 보존 및 활용을 위한 지침 작성 지원								100%
		사업관리								100%
		.월간업무보고								100%
		.최종보고서작성								100%

3. 국내·외 기술개발 현황

3.1. 국내 기술개발 현황

○ 중앙부처 기록관리시스템 (CRMS : Cloud Record Management System)

- 중앙부처가 직접 구축하고 운영하며, 2017년 15개 기관을 시작으로 2018년 27개 기관, 2019년 5개 기관으로 확산 예정



<그림 86> CRMS 기록관리시스템

○ 핸디소프트 HANDY AMS

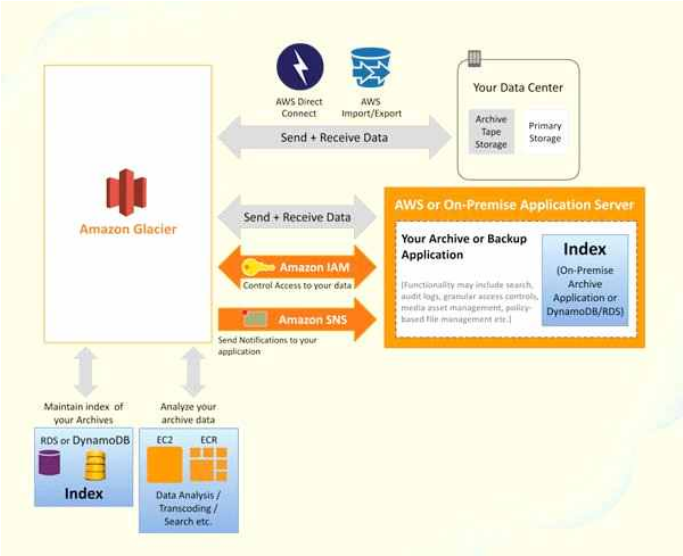
- 기록물의 진본성, 무결성, 신뢰성, 이용가능성을 보장하며, 관련 법령의 완벽 준수는 물론 인프라 공동 이용, 단계적 사업 추진으로 효과적인 시스템 구축이 가능

<그림 87> 핸디소프트 HANDY AMS

3.2. 국외 기술개발 현황

○ Amazon Web Services (AWS) Glacier

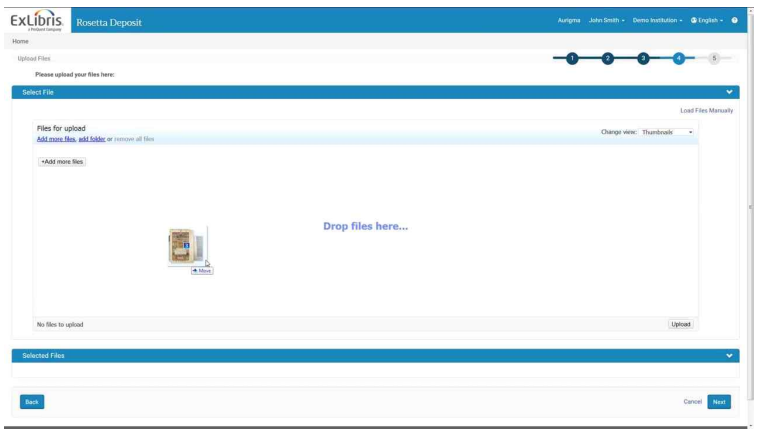
- 기존 S3 호환 애플리케이션, 도구, 코드 ,스크립트 및 수명 주기 규칙으로 활용 가능
- SSL을 통한 데이터 전송을 지원, 자동으로 데이터 암호화, AMazon IAM 서비스를 사용하여 데이터 액세스 제어 기능
- 정기적이고 체계적인 데이터 무결설 점검 및 자가 치유 기능



<그림 88> Amazon Web Services (AWS) Glacier

○ Ex-Libris Rosetta

- 데이터 일관성 유지
- API, 플러그인 및 개방형 인터페이스 제공



<그림 89> Ex-Libris Rosetta

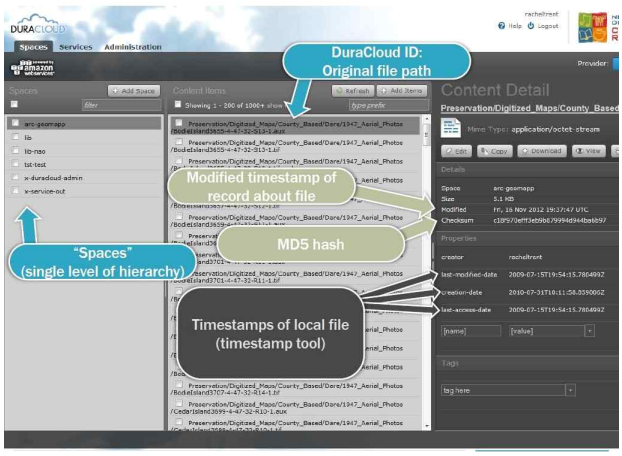
○ preservica (프리저비카)

- OAIS(Open Archival Information System) 개념 기반 기술
- AWS, Azure Cloud를 기반으로 서비스
- 전자기록 입수, 보존, 서비스가 가능
- 정보패키지에 전자서명, 해시값 등을 포함하여 무결성을 보장
- 유지보수가 어려움

<그림 90> preservica (프리저비카)

○ Duracloud (듀라클라우드)

- OAIS(Open Archival Information System) 개념 기반 기술
- Desktop App, REST API, 웹 인터페이스 기능
- 최소 1년에 2번 모든 콘텐츠에 대한 비트 무결성 상태 보고서 기능



<그림 91> Duracloud (듀라클라우드)

Provider / Product	Choice of Locations	Speed of Access	Degree of Adoption	Costs	Security	Data Migration Out
Amazon Web Services (AWS) Glacier	EEA (Ireland) and Global	Typically within 3-5 hours	High	No initial costs. Billed for usage by the hour	Comprehen sive accreditatio ns	Download standard formats by API, and move large data volumes on disk
Amazon Web Services (AWS) S3	EEA (Ireland) and Global	Immediate, by widely adopted API	Very High	No initial costs. Billed for usage by the hour	Comprehen sive accreditatio ns	Download standard formats by API, and move large data volumes on disk
CloudSigma	Switzerland (EEA equivalent) and USA	Immediate, by API	High	No initial costs. Billed for usage in 5 minute increments	Suitable accreditatio ns, some throught hosting provider Interaxion	Download standard formats by API
GreenQloud	EEA (Ireland) and USA	Immediate, by AWS compatible API	High	No initial costs. Billed for usage by the hour	Suitable accreditatio ns, some through hosting partner Verne Global	Download standard formats by API
Microsoft Windows Azure	EEA and global	Immediate, by API	High	No initial costs. Billed for usage by the minute	Comprehen sive accreditatio ns	Download standard formats by API. USoption to move large data volumes on disk not yet available in Europe
Rackspace	UK and Global	Immediate, by Open Stack API	High	No initial costs. Billed for usage by the hour	Comprehen sive accreditatio ns	Download standard formats by API

<표 103> 일반 클라우드를 활용한 기록관리 사례

Provider / Product	Choice of Locations	Speed of Access	Degree of Adoption	Costs	Security	Data Migration Out
Arkivum A-Stor	UK data centres	Access to tape-based storage, typically within 5 minutes of request by file system, API, or GUI	Moderate	Annual subscription, or paid-up fixed term contacts, based, upon volume and duration	Certified to ISO27001 and audited on a 6 monthly basis	Company offers comprehensive escrow arrangement
Arkivum OSCAR	installed locally	Access to tape-based storage, typically within 5 minutes of request by file system, API, or GUI	Moderate	Annual subscription, or paid-up fixed term contacts, + hardware	Depends upon accreditations at host institution's data centre	Can have two Pods and an offline copy. Pods can be managed locally (or remotely by Arkivum)
DuraSpace DuraCloud	AWS data centres in USA	Immediate, by AWS API	Moderate	Annual subscription, based upon volume	As AWS Comprehensive accreditations	Source records available for retrieval by API. Client can opt to sync to Rackspace as 2nd cloud service to AWS
Internet Archive Archive It	USdata cenres	Immediate access by web User Interface	Moderate High. Over 300 partners in 16 countries	Annual subscription, based upon volume	No formal accreditations?	Partner Institutions can receive a copy on a hard drive or download their files directly from servers
Preservica Cloud Edition	AWS data centres in EEA (Ireland) and US	Immediate, by AWS API	Moderate High. Over 300 partners in 16 countries	Annual subscription, based upon volume	As AWS Comprehensive accreditations	Source records available for retrieval by API or user can copy home to
Archivematica	Open source preservation software - cloud hosting of the software is being tested in Wales in combination with cloud-based archival storage. Cloud hosting of the software is operational in Canada					
Ex-Libris Rosetta	Rosetta digital preservation system not currently available as a cloud installation but cloud product release is under review					

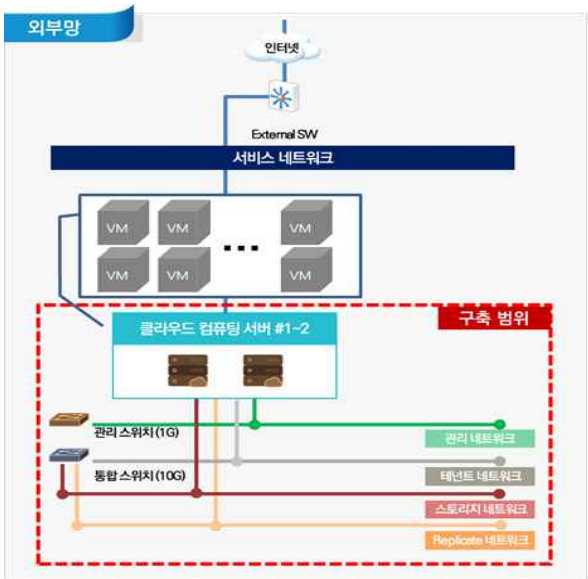
<표 104> 기록관리용 특수 클라우드를 활용한 기록관리 사례

제2장 제2세부연구개발과제의 연구개발 내용 및 방법

1. 연구 대상

1.1. 보존방식에 따른 기술적합도 검증을 위한 테스트베드 구축

○ 에뮬레이션 시험 검증을 위한 하드웨어 시스템 구축

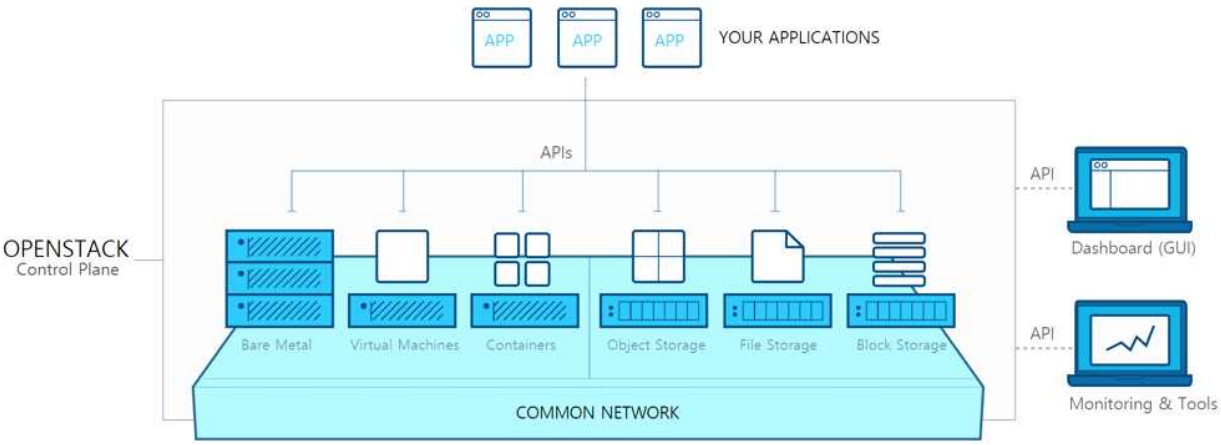


예정 장비 스펙

구분	사양	수량	비고
클라우드 컴퓨팅 서버	• CPU: 24C/48T • RAM: 256GB • DISK1: SSD 480GB * 2EA • GlusterFS DISK: 1TB * 6EA	2대	GlusterFS
통합 스위치	• 10G 12Port Ethernet	1대	
관리 스위치	• 1G 8Port Ethernet	1대	

<그림 92> 에뮬레이션 시험 검증을 위한 하드웨어 시스템 구축

○ 오픈스택(OpenStack)기반 클라우드 인프라 환경 구축



<그림 93> 오픈스택 기본 구조

- 시험환경을 고려하여 기본 오픈스택 프로젝트 설치

프로젝트명	서비스 내용	비고
glance	· 이미지 서비스	
horizon	· 포탈 서비스	
keystone	· 인증서비스	
neutron	· 네트워크 서비스	
nova	· 컴퓨트 서비스	
heat	· 오케스트레이션 서비스	
magnum	· docker 서비스	

<표 105> 오픈스택 주요 컴포넌트

- 오픈스택 기반 클라우드 인프라의 운영·관리를 위해 오픈스택잇(Openstackit) 포털 설치

오픈스택잇 UX 특징

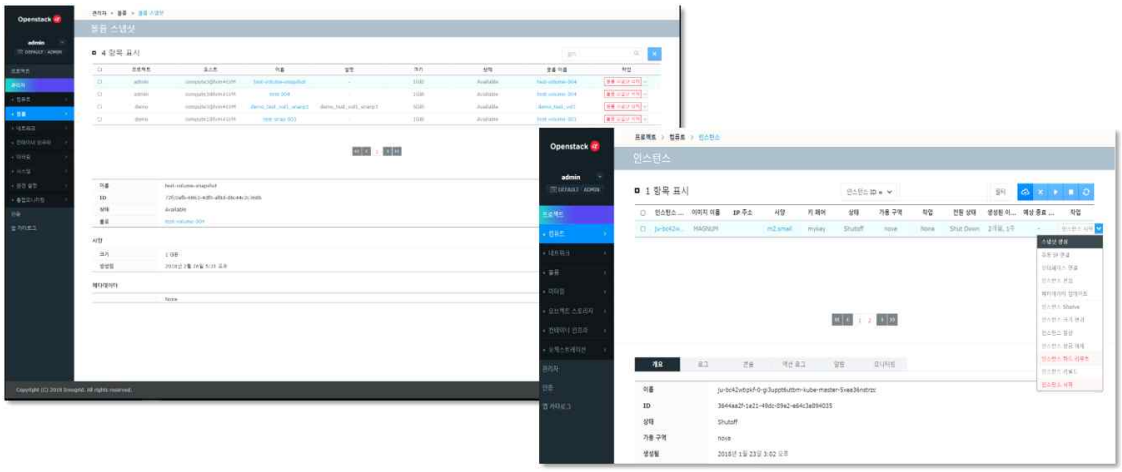
- 클라우드잇 디자인 노하우에 기반한 UX(사용자 경험) 극대화
- 위자드를 통한 자원 생성 화면 제공
- 목록과 상세내용 동시 표현
- 화면 이동을 최소화한 구성
- 관리자/사용자 권한에 따른 서로 다른 기능 및 화면 구성 제공
- 사용자를 위한 기능을 별도로 제공하는 셀프 서비스 포털 제공

<그림 94> 오픈스택잇 포탈 특징

1.2. 데이터세트 유형 전자기록의 에뮬레이션 시험

○ 선정된 데이터세트의 에뮬레이션 보존 방식 테스트

<그림 95> 클라우드 기반 가상화 환경에서의 에뮬레이션 환경 구성



스냅샷 기능

- 인스턴스, GPU 인스턴스, 볼륨 데이터의 시점 백업 기능인 스냅샷 기능 제공

볼륨 백업 기능

- Full Backup
- Incremental Back up (증분 백업)

<그림 96> 가상화 기반 에뮬레이션 환경의 스냅샷 및 백업/복원 시험 검증

○ 선정된 데이터세트의 에뮬레이션 후 원천 데이터세트와의 정합성 검증항목 마련 및 점검

- 데이터세트 유형, 환경 등 특성에 따라 에뮬레이션 보존 방식 적합 모델 도출
- 데이터세트 유형, 규모, 시스템 환경에 따라 에뮬레이션 전후 정합성 검증 항목 도출

2. 연구 방법

2.1. 보존방식에 따른 기술적합도 검증을 위한 테스트베드 구축

○ 에뮬레이션 시험 검증을 위한 하드웨어 시스템 구축

- 에뮬레이션 시험 검증 요구사항에 따라 클라우드 인프라 규모 사이징, 하드웨어 아키텍처 및 네트워크 구성(안) 수립
- 기존 클라우드 인프라 구축 경험을 통해 안정성이 검증된 하드웨어 장비 확보
- 참여기업들 간의 원활한 연구개발 추진을 위해 하드웨어 시스템을 민간 데이터센터에 구축

○ 오픈스택 기반 클라우드 인프라 환경 구축

- 안정성이 검증된 오픈스택 퀸즈 버전으로 클라우드 인프라 환경 구축
- 에뮬레이션 방식의 전자기록물 보존 시험·검증을 위해 가상머신, 도커/컨테이너 환경 제공 수준의 클라우드 인프라 환경 구축
- 이를 위해, 다양한 오픈스택 프로젝트 중 불필요한 프로젝트를 제외한 최적의 프로젝트들만 설치

2.2 데이터세트 유형 전자기록의 에뮬레이션 시험

○ 선정된 데이터세트의 에뮬레이션 보존 방식 테스트

- 공공기관의 테스트용 데이터세트 선정을 위한 과제수행부서의 기관 방문, 면담 등을 지원하고, 선정된 데이터세트 운영 현황 분석을 통해 보존 범위(OS, 데이터셋, 애플리케이션 등)를 정의
- 클라우드 테스트베드 기반 가상화 환경에 데이터세트 및 운영 환경을 마이그레이션하고, 클라우드의 스냅샷, 백업 기능을 통해 가상화 이미지로 보존
- 보존된 가상화 이미지로부터 복원 후 데이터 세트 정합성 검증

○ 선정된 데이터세트의 에뮬레이션 후 원천 데이터세트와의 정합성 검증항목 마련 및 점검

제3장 제2세부연구개발과제의 최종 연구개발 결과

1. 데이터세트유형 전자기록 현황 및 장기보존 기술 조사

- 데이터세트 유형의 전자기록 관리 전략중 하나인 에뮬레이션 사례에 대하여 조사함
- 디지털 양피지, 올리브 프로젝트, 디지털 포렌식에서의 에뮬레이션 사례에 대하여 조사함
- Unix 계열 가상화 기술과 대안, Unix to Linux 전환 방법 및 고려사항, 전환 사례에 대하여 조사함

1.1 에뮬레이션 사례 조사

- 디지털 양피지(Digital Vellum)는 모든 소프트웨어 하드웨어가 포함된 데이터 인프라 자체를 디지털 형태로 클라우드 서버에 보존하는 기술을 말함
- 올리브(OLIVE) 프로젝트는 대표적인 클라우드 컴퓨팅 기반 에뮬레이션 기법으로, 가상화 머신(VM) 기술을 도서관 아카이브에 적용한 사례
- 범죄 수사를 위해 디지털 장비 분석 등을 하는 디지털 포렌식에서의 에뮬레이션 사례는 안드로이드 장치에 대한 사례가 대부분
- 독일의 bwFLA 프로젝트는 클라우드 컴퓨팅 환경 기반의 에뮬레이션 기법을 구현하여 에뮬레이션 전략을 서비스로서 제공하는 것을 목표로 하며, 국외에서는 클라우드 컴퓨팅 환경에서의 에뮬레이션 서비스를 구축하고 제공하기 위한 연구 및 개발이 상당한 진전이 있음

○ 디지털 양피지(Digital Vellum)

- ‘아래아한글’ 소프트웨어를 개발하던 ‘한글과컴퓨터’가 갑자기 폐업하여 아래아 한글 파일을 읽어 들일 수 있는 소프트웨어는 사라질 수 있음. 이러한 상황이 온다면 아래아 한글로 작성된 모든 전자문서는 사라지게 될 가능성이 높음

- 디지털 기록물을 영구히 보존하기 위해 가장 먼저 생각나는 해결책은 모든 소프트웨어가 개방된 표준 파일 포맷을 도입하는 것으로 누구나 자유롭게 해당 포맷을 읽어 들일 수 있고 관련 기술을 구축할 수 있도록 한다면 ‘기록의 상실’은 후대에도 일어나지 않을 확률이 높다고 생각할 수 있음

- 획일적인 표준 디지털 포맷의 소프트웨어는 구속력이 떨어져 상업적 매력이 크게 감소하기 때문에 이윤을 최우선으로 추구하는 기업들은 인류 기록의 보존을 위해 표준 포맷을 고수하지 않을 것임

- 파일 형태의 문서가 있더라도 구동하는 소프트웨어가 존재하지 않는다면 해당 파일은 결국 ‘존재하지 않는’ 정보가 됨. 그래서 문서 파일과 함께 소프트웨어도 보전해야 함

- 구글의 부사장인 빈트서프(Vinton Gray Cerf)가 2015년 2월에 제안한 디지털 양피지는 디지털 기록물을 영구히 보존할 수 있는 기술로서 모든 소프트웨어와 하드웨어, 즉 데이터 인프라 자체를 디지털 형태로 클라우드 서버에 보존하는 기술임

○ 올리브(OLIVE) 프로젝트: 미국 카네기 멜론 대학교, IBM연구소가 공동 추진

- 대표적인 클라우드 컴퓨팅 기반 에뮬레이션 기법

- 디지털 양피지와 유사한 기술은 미국 카네기멜론대학교와 IBM연구소가 공동으로 추진하고 있는 '올리브'(OLIVE) 프로젝트임

- 올리브는 Windows3.1 이나 초기 맥(MAC)컴퓨터의 환경을 가상화 기술로 이용하여 응용프로그램, 디지털 문서 파일, 운영 체제까지 포함한 "가상화 머신 이미지 또는 템플릿 이미지"들을 만들고 클라우드 컴퓨팅 환경 내부에 설치하여 누구나 외부에서 클라우드 컴퓨팅 환경에 접속하여 디지털 문서 파일을 원격으로 열람할 수 있는 서비스를 제공

- 다음의 <그림 97>은 올리브 시스템 내에서 어떻게 콘텐츠들(OS, 응용SW 등)이 추가되고 운용되는지를 나타냄

- 클라우드 컴퓨팅 기술로 구현된 "VM Image Library"에 사용자들이 OS와 해당 OS에서 동작하는 응용SW(Pac-Man)를 설치 및 업로드

- 다른 사용자는 응용SW가 탑재된 OS를 클라우드 컴퓨팅 환경에 설치하여 응용SW를 활용할 수 있는 구조를 통해 서비스를 제공

- 올리브는 가상화 머신(VM) 기술을 도서관 아카이브에 적용한 사례로, IBM은 가상화 기술이 모든 곳에 편재되고 대규모 가상화 데이터센터가 부상할 것을 예측하고 2006년부터 관련 기술 개발에 주력해 왔음

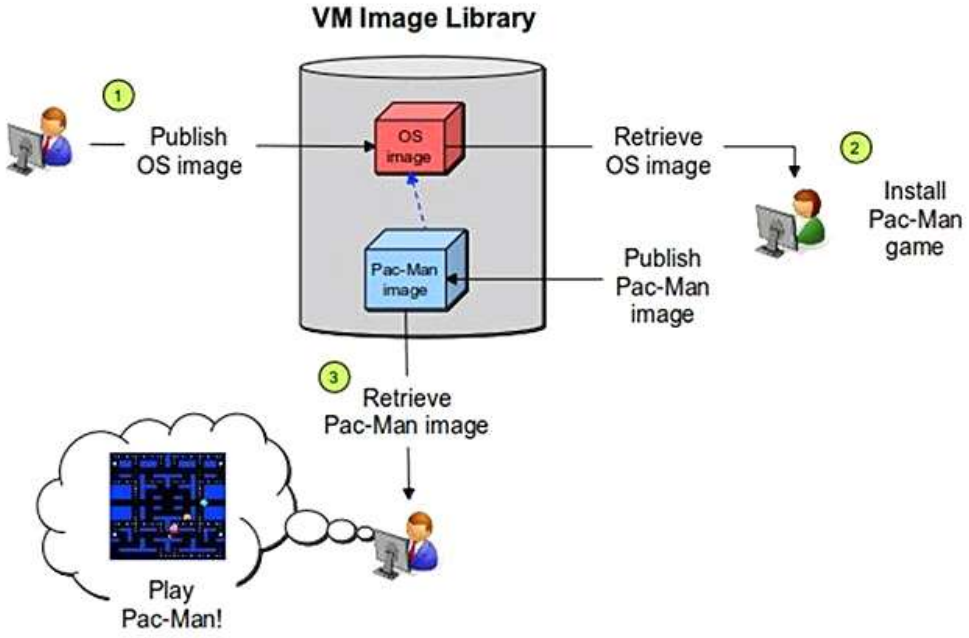
- 가상화 이미지를 스트리밍하는 기술을 개발한 마하디프 사티아나라이아난 카네기멜론대 교수와 IBM연구소가 협업하여 진행되었음

- 2011년 디지털 도서관 프로젝트에 적용되어 올리브라는 프로젝트가 탄생하였음

- 올리브는 '가상화 실행을 위한 열린 이미지 도서관'(Open Library of Images for Virtualized Execution)을 의미하는 영문의 약자로 가상화 기술을 활용한 퍼블릭 도메인 도서관 프로젝트로 올리브는 다양한 문서 형태로 저장돼 있는 연구자료를 모아 가상화 기술 위에서 누구나가 실행할 수 있도록 지원함

- 올리브 프로젝트에서는 가상화 머신 이미지인 '소프트웨어 X레이 스냅샷'을 사용하였고, 이 이미지 안에는 디지털 문서 파일을 비롯해 응용 소프트웨어, OS, 프로그램 기능 코드 등을 모두 담을 수 있는 플랫폼임

- 이 플랫폼은 클라우드 컴퓨팅 기술로 구현될 수 있음



<그림 97> 올리브의 기술 구현 매커니즘

○ 디지털 포렌식에서의 에뮬레이션

- 디지털 포렌식은 범죄 수사를 위해 디지털 장비 분석 등을 하여 증거를 수집하는 행위 등을 통칭
- 디지털 포렌식 종류
- 컴퓨터 법과학: USB 드라이브, SD 드라이브 등등 복원
- 모바일 장치 법과학: 내장 된 GPS/ 위치추적 또는 셀 사이트로그 범위 추적, 내장된 통신 시스템(예: GSM)
- 네트워크 법과학: 정보수집 및 로컬 및 WAN/인터넷의 네트워크 트래픽을 모니터링 하고 분석 패킷레벨분석법
- 데이터 분석 법과학: 금융 범죄로 인한 사기 행위 패턴을 발견 분석 구조화 된 데이터 조사데이터베이스 법과학: 데이터베이스와 관련 된 포렌식 / 인로그, 데이터베이스 내용. RAM의 타임라인 구축 및 복구
- 디지털 포렌식에서 에뮬레이션 방식의 사용은 대부분 스마트폰(안드로이드 장치에 대한 에뮬레이션)과 관련된 사례가 대부분
- 디지털 포렌식 과정에서 발생한 증거물에 대한 장기 보존 사례는 찾기 어려웠음
-

○ bwFLA(Baden-Wuerttemberg Functional Long-term Archiving and Access) 프로젝트: 독일

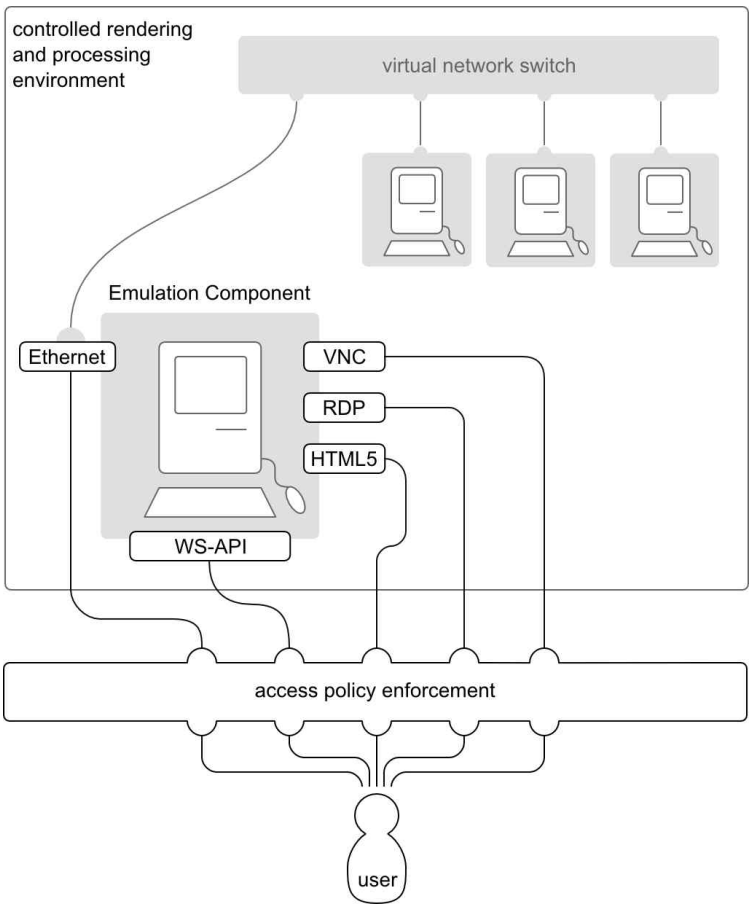
- 목표: 클라우드 컴퓨팅 환경 기반의 에뮬레이션 기법을 구현하여 에뮬레이션 전략을 서비스(EaaS: Emulation As A Service)로서 제공

- 복잡한 디지털 기록이 지닌 특성을 장기보존 할 수 있으며 본래의 기능 및 모습을 경험하기 위한 가장 좋은 방법은 생성 당시의 응용 프로그램을 사용하는 것이라는 명제에서 출발함

- HW와 OS의 가상화 및 클라우드 컴퓨팅을 통해 구형 응용 프로그램을 실행시킬 수 있으며 일반인에게도 쉽게 접근할 수 있도록 구현함

- MAAS/Juju, OpenStack로 클라우드 컴퓨팅 환경을 구축, PPC, m68k, Intel-based x86등의 주요 CPU 구조와 OS/2, MS Windows, MacOS 등 주요 OS를 지원함

- 아래의 <그림 98>처럼 사용자에게는 VNC, RDP, HTML5, WS-API, Ethernet 등을 통해 인터넷으로 접근 서비스를 제공



<그림 98> bwFLA 시스템 구조와 서비스 제공

○ 국외에서는 클라우드 컴퓨팅 환경에서의 에뮬레이션 서비스를 구축하고 제공하기 위한 연구 및 개발을 통해 상당한 진전이 이루어짐

○ 클라우드 컴퓨팅 기술이 성숙되어 있는 현재 시점에서 국내에서도 가능한 신속하게 클라우드 컴퓨팅 기반 에뮬레이션 전략을 활용한 전자기록 장기보존 방안을 연구하는 것이 필요하다고 생각됨

1.2 U2L 사례 조사

- x86 기반 가상화 제품들은 유닉스 서버를 지원하지 않음에 따라 UNIX 계열 가상화와 대안에 대한 사례를 조사함
- Unix 기반 시스템을 x86의 Linux 기반 시스템으로 전환하는 방식인 UNIX to Linux(U2L) 의 전환 절차와 고려사항에 대하여 조사함
- 현업에서 사용한 U2L 전환 사례, 방식과 기간에 대하여 조사함

1.2.1 UNIX 계열 가상화 및 대안

○ x86 CPU 기반으로 운용되는 거의 대부분의 운영체제는 클라우드 컴퓨팅 기반 에뮬레이션이 가능하나, UNIX 계열의 운영체제를 에뮬레이션 하는 것은 불가능한 것으로 조사됨 (HP-UX 등의 UNIX 계열)

○ 클라우드 컴퓨팅 기술(가상화 컴퓨팅 기술)은 x86 가상화를 기반으로 이루어지고 있음

- 주요 x86 기반 가상화 제품은 VMware의 vSphere, Citrix의 Xen Server, Microsoft사의 Hyper-V, Linux 기반의 KVM 등이 있음
- x86 기반 가상화 제품들은 유닉스 서버를 지원하지 않음 (CISC와 RISC 칩의 설계 방식이 다르기 때문)

○ UNIX 계열의 운영체제는 현재 시장이 죽어가고 있기 때문에 “IT 기술 및 자본”이 이를 외면하고 있는 추세

○ 전자기록 장기보존에는 가장 발전되고 안정적인 IT 기술을 활용하여야 하는데, IT 자본이 외면하고 있는 IT 기술이 성숙되기를 기다릴 수는 없다고 판단

○ UNIX 및 응용 프로그램을 클라우드 컴퓨팅 기반으로 “그대로” 에뮬레이션 하는 것은 클라우드 컴퓨팅 전문가로써 불가능하다고 판단

○ 운영체제 자체를 에뮬레이션 하는 것이 아니라, 다른 운영체제로 대체하고 유사한 응용프로그램을 실행하는 방법을 생각 할 수 있음

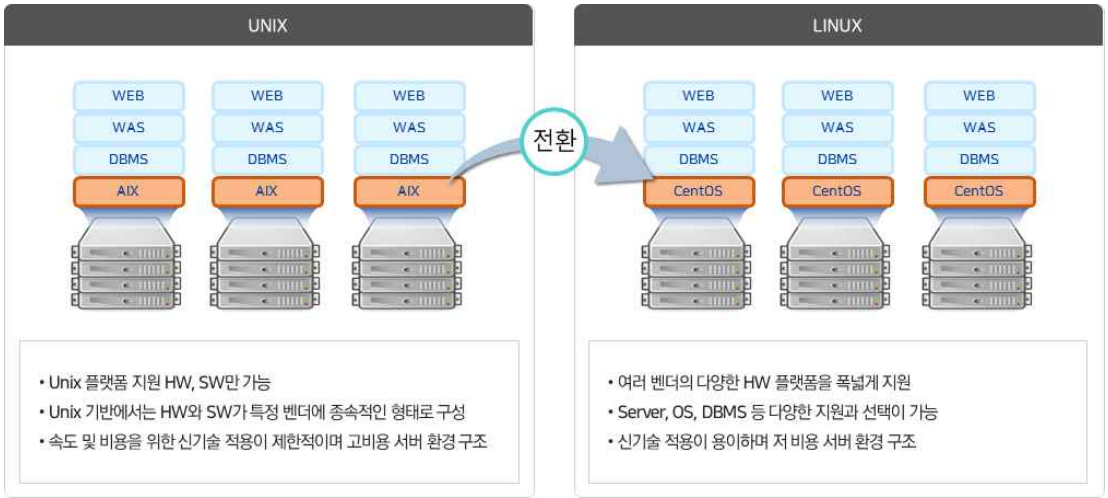
○ UNIX 계열의 운영체제는 Linux로 대체, 동일한 응용프로그램을 활용하여 원본과 동일 기능 및 유사한 외관을 보여줄 수 있는 기술(U2L)은 “고비용이지만” 가능하기 때문에 필요하다면 도입 할 수 있음

○ 본 과제의 목표는 데이터세트의 기능과 외관을 장기 보존하는 것으로 데이터 세트 및 DBMS 가 운용되는 운영체제가 에뮬레이션이 불가능하다면, U2L 방식을 사용하는 방안이 “클라우드 컴퓨팅 기반 에뮬레이션”의 좋은 대안으로 판단됨

1.2.2 U2L 전환 개요

○ U2L (Unix To Linux) : Unix 기반 시스템을 x86의 Linux 기반 시스템으로 전환하는 방식

○ 성능 및 안정성 측면에서 Unix가 더 뛰어나지만 비용대비 성능 측면에서 보았을 때 Linux를 사용하는 것이 더 효율적임



<그림 99> Unix vs. Linux

1.2.3 U2L 전환 프로세스 및 고려사항

<그림 100> U2L 전환 프로세스

항목	설명
개발 환경	· 컴파일 및 링커 옵션이나 make 도구의 차이로 인한 Makefile에서의 수정이 필요 · C 정수데이터의 경우 Endian 변환 필요
커널 튜닝	· Unix 커널에 적용된 OS 튜닝항목에 대한 조사 후 RHEL에서 맵핑하여 적용
보안	· 사용자 접근 제어, 침입 탐지, 로컬 시스템, 방화벽과 같은 보안 설정에 대한RHEL 적용, TCP Wrapper, AIDS, iptables 사용
파일 시스템	· 기존 JFS,UFS,VxFS 파일 시스템은 RHEL에서 연결해서 사용불가. 데이터 이전 고려 필요
디버깅 및 프로파일링 도구	· RHEL에서 제공되는 system, oProfile,ftrace 등의 디버깅도구에 대한 지식 습득
소프트웨어 라이프사이클 관리	· RHN Satelite과 같은 SW 패키지 관리를 제공해주는 프로비저닝/ 업데이트/ 관리시스템 환경 구축 및 기존 솔루션 활용 가능여부 확인
가상화	· 시스템 로드가 크지 않는 경우 1:1 단일서버로의 이전보다는 가상화 기술을 사용하여 통합이전을 고려하는 것이 비용 효율적
3rd-party 라이브러리	· 미들웨어와 같은 3rd-Party 라이브러리를 사용하는 경우 RHEL에서 동일버전이나 호환가능한 버전이 인증되어 있는지 확인 필요

<표 106> U2L 전환 고려사항

○ 기존 시스템의 현황 및 규모에 따라 매우 가변적이지만, U2L을 위해 전문화된 Toolkit, 조직/운영 구성 필요

○ 애플리케이션, 데이터량, 마이그레이션 난이도에 따른 높은 전환 기간 발생 여부 검토

- 애플리케이션: U2L을 위한 사전테스트 및 소스코드 수정 등의 기간 확보 필요
- 데이터: 실제 마이그레이션 수행 방법에 따라서 소요 기간 다수 발생 (예: Export/import, Oracle TTS와 같은 방법을 사용하는 경우, 데이터량에 따라 다운타임이 결정됨)
- DB전환: 기존 Oracle, DB2에서 MariaDB, PostgreSQL 등으로 전환하는 것이므로 DB 종속성 검증을 위한 기간 확보 필요

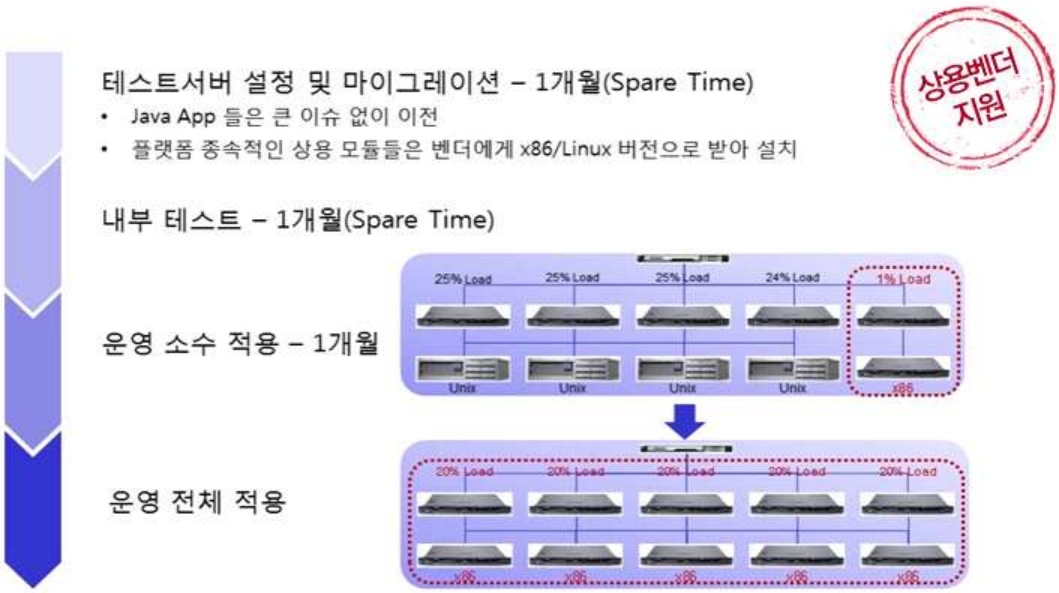
○ Unix 특화 애플리케이션 유/무, 특정 기능의 마이그레이션 지원 불가 등 신규개발 가능성 여부 검토

- Unix 종속적인 라이브러리를 사용하는 애플리케이션 사용 여부 검토 필요
- Unix 시스템 또는 메인프레임에서만 구동되는 특수 업무 존재 여부 검토 필요
- 자체 개발된 솔루션의 소스 보유 유/무 검토 필요 (플랫폼 변경에 따른 개발 불가능 상황 발생 가능)

1.2.4 U2L 전환 사례

○ GS-Shop의 Main App 서버를 기존 Unix/상용솔루션에서 Linux/오픈소스 + 가상화까지 적용하여 전환한 사례

- JAVA 이관에 대한 이슈보다는 외부 모듈들을 다시 컴파일 하거나 다시 Linux 장비에 맞는 모듈을 Vendor에서 제공받는데 걸리는 시간이 좀 더 큼
- 소스코드가 얼마나 JAVA로 구성되어 있는지에 따라, 외부 모듈을 사용하지 않느냐에 따라 U2L 전환 난이도 판별



<그림 101> U2L 사례 1: GS-Shop App 서버 전환

○ EC(E커머스) 분석 - Oracle DB서버를 Linux서버로 전환한 사례

- 튜닝 등 외부 모듈(ex. pro-C)들을 다시 컴파일 하는 경우 전문 업체를 미리 확보한 후 프로젝트 진행하는 것이 효율적임

<그림 102> U2L 사례 2: EC분석 DB 서버 전환

2. 기술적합도 검증을 위한 테스트베드 구축

- 에뮬레이션 전환의 기술적합도 검증을 위한 테스트베드를 구축함
- 테스트베드 구축을 위한 하드웨어 시스템을 도입하고, 클라우드 환경 및 에뮬레이션 시험 환경을 구축함

2.1 하드웨어 시스템 도입 및 구성

- 테스트베드 구축을 위한 하드웨어 시스템의 인프라 규모를 사전 정의하여 설계하였고 서버 2대, 스위치 2대로 구성하여 이노그리드 서버실에 구축함

○ 다양한 클라우드 인프라 구축 노하우를 토대로 클라우드 환경 제공을 위한 하드웨어 시스템 도입

- 클라우드 환경 구축을 위한 하드웨어 시스템 인프라에 대한 사전 사이징을 <표 107 ~ 109>과 같이 진행하고, <그림 103>와 같이 아키텍처를 설계함

대상 서버	CPU 코어 (개)	CPU 클럭 (GHz)	CPU 사용률 (%)	VM 대수	필요 CPU 클럭 (GHz)
에뮬레이션 서버1	8	2.6	70	3	52.4
에뮬레이션 서버2	4	2.6	50	2	12.5
에뮬레이션 서버3	4	2.6	50	2	12.5
필요 CPU 클럭 합계	64.5 GHz				
오버헤드 (+ 20%)	77.4 GHz				
기준 호스트 대수 (÷2대)	38.7 GHz				
HV사용보정 (+ Host Clock*2)	48.9 GHz				
MGR 사용보정 (+ Host Clock*2)	49.1 GHz				
GlusterFS 사용보정 (+ Host Clock*2)	54.3 GHz				
적합 CPU 모델	Intel® Xeon® Gold Processor 6126 (62.4GHz /Host)				
산정근거	(전체 서버의 필요 CPU 클럭 합계+오버헤드)÷기준 호스트 대수 + CPU 보정 · 필요 CPU 클럭 = (CPU 코어 수 X CPU 클럭 X CPU 사용률) 합계 · 오버헤드: 확장성 및 사용률 최대치를 고려한 여분 클럭(20%) · 기준 호스트 대수는 2대로 산정 · 하이퍼바이저 사용 CPU 보정 (+호스트 CPU 클럭 * 2Core) · 오픈스택잇 MGR CPU 보정 (+호스트 CPU 클럭 * 2Core) · 분산파일시스템(GlusterFS) CPU 보정 (+호스트 CPU 클럭 * 2Core)				

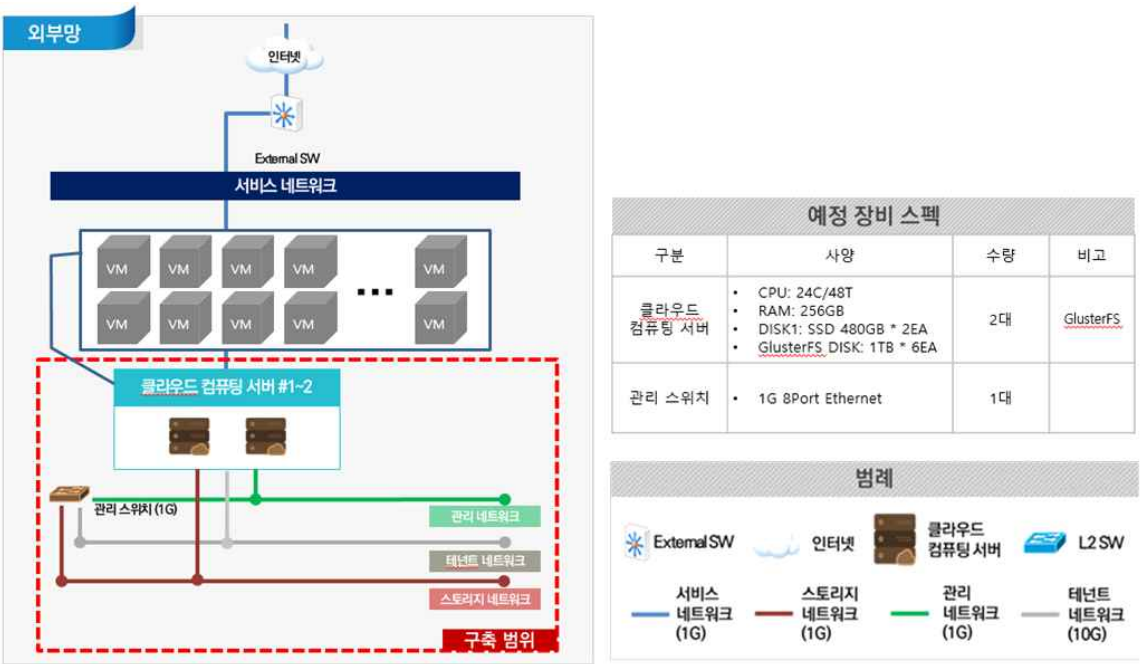
<표 107> 인프라 사전 사이징: 물리 호스트 1대 당 필요 클럭 산정

CPU 모델명	CPU 코어 (개)	CPU 클럭 (GHz)	호스트당 CPU	호스트당 클럭(GHz)	호스트 대수	제공 총 클럭	적합성
Gold 6126	12	2.6		62.4		124.8	적합
Gold 5115	10	2.4	2	48	2	96	부족
Silver4116	12	2.1		50.4		100.8	부족

<표 108> 인프라 사전 사이징: 클라우드 서버 CPU 모델 선정

대상 서버	메모리 용량 (GB)	평균 메모리 사용률 (%)	VM 대수	필요 메모리 용량 (GB)
에뮬레이션 서버1	64	100	3	192
에뮬레이션 서버2	16	100	2	32
에뮬레이션 서버3	16	100	2	32
필요 메모리 용량합계	256 GB			
오버헤드 (+ 20%)	307.2 GB			
기준 호스트 대수 (÷2대)	153.6 GB			
HV사용보정 (+32GB)	135.6 GB			
MGR 사용보정 (+32GB)	217.6 GB			
GlusterFS 사용보정 (+4GB)	221.6 GB			
적합 메모리 용량	256 GB /Host			
산정근거	(전체 서버의 필요 메모리 용량 합계 + 오버헤드) ÷ 기준 호스트 대수 + 하이퍼바이저 사용 메모리 보정필요 메모리 클럭 = (메모리용량 X 평균 메모리 사용률)합계 · 오버헤드: 확장성 및 사용률 최대치를 고려한 여분 용량(20%) · 기준 호스트 대수는 2대로 산정 · 하이퍼바이저 사용 메모리 보정 (+호스트 1대당 32GB) · 오픈스택잇 MGR 메모리 보정 (+노드 1대당 32GB) · 분산파일시스템(GlusterFS) 메모리 보정 (+노드 1대당 4GB)			

<표 109> 인프라 사전 사이징: 메모리 리소스 설계



<그림 103> 하드웨어 시스템 인프라 아키텍처 설계

- 위와 같이 사전 인프라 사이징을 하였으나 예산 및 기타 제약 사항으로 인해 실 도입 장비는 <표 110>과 같음

구분	카테고리	상품명	수량
서버 (2대)	Intel R2312WFTZS	Intel® R2308WFTZS 2U 8Bays 1+1 Dual Power System	2
		Intel® 2U Server System R2308WFTZS	1
		Intel® Xeon® Silver Processor 4116 (12Core, 2.1GHz/85W)	2
		16GB DD4 ECC Registered DIMM 19200(2400)	16
		Intel® SSD 480GB, 2.5in(S4510시리즈)	2
		1TB SAS 12Gb/s 7.2K RPM (128MB)	4
		Intel® Integrated RAID Module RMS3CC080 8 internal ports	1
		Intel® Ethernet Server Adapter I350-T4V2(1G UTP/Quad)	2
		Intel® 1300w AC 80+ Titanium Efficiency Power supply AXX1300TCRPS	1
		AXXERVRAIL Basic Rack Rail	1
스위치1 (1대)	JH295A	HPE 1950 12XGT 4SFP+ Switch	1
	U8PP6E	HPE 1Y FC NBD Exch 1950 24G-2XGT2SFP SVC [for JH295A]	
스위치2 (1대)	J9776A	Aruba 2530 24G Switch	1
	H1GS8E	HPE 1Y FC NBD Exch Aruba 2530 24G Sw SVC [for J9776A]	

<표 110> 테스트베드 하드웨어 도입 장비 (*발주서 참고)

- 도입된 테스트베드의 하드웨어 시스템을 <그림 104>과 같이 구성하였으며, 현재 <그림 105>와 같이 이노그리드 사내 서버실에 설치 및 구축됨 (사업 종료 후 국가기록원 이전 예정, 일정은 미정)

<그림 104> 하드웨어 시스템 구성도

<그림 105> 테스트베드 하드웨어 시스템 설치 환경 (이노그리드 사내 서버실)

- 서버1은 관리 및 컴퓨팅 서버, 서버2는 컴퓨팅 서버로 용도를 구분하고 각 서버의 하드웨어 설정을 <표 111> ,<표 112>와 같이 구성하였으며, 전체 시스템의 사양은 <표 113>과 같음

구분	항목	설정 내용
BIOS	BIOS MODE	Legacy Mode
	Hyper-Threading	Enable
	CPU Virtualization	Enable
	CPU Power Profile	Power Mode
RAID	SSD Volume 480GB x2	RAID 1 = 480GB
	SAS Volume 1TB x2	RAID 0 = 2TB

<표 111> 서버 1 (관리 및 컴퓨팅 서버) 하드웨어 설정 정보

구분	항목	설정 내용
BIOS	BIOS MODE	Legacy Mode
	Hyper-Threading	Enable
	CPU Virtualization	Enable
	CPU Power Profile	Power Mode
RAID	SSD Volume 480GB x2	RAID 1 = 480GB
	SAS Volume 1TB x6	RAID 0 = 5TB

<표 112> 서버 2 (컴퓨팅 서버) 하드웨어 설정 정보

항목	CPU	메모리	볼륨스토리지	이미지스토리지	가상서버스토리지
서버1	24Core	256GB	1TB	1TB	3TB (공유 스토리지)
서버2	24Core	256GB	-	-	
총합	**48Core**	**512GB**	**1TB**	**1TB**	**3TB**

<표 113> 전체 시스템 사양 정보 요약

2.2 클라우드 환경 및 에뮬레이션 시험 환경 구축

- 오픈소스 활용을 위해 오픈스택(Openstack) 퀸즈(Queens) 버전으로 클라우드 환경을 구축함
- 클라우드 인프라의 운영 및 관리를 위한 이노그리드 포털 솔루션인 오픈스택잇(Openstackit)을 설치함
- 에뮬레이션 시험을 위해 필요한 오픈스택잇 기능 항목을 도출하고 검증함

○ 오픈소스 활용을 위해 오픈스택(Openstack) 기반의 클라우드 환경 구축

- 에뮬레이션 시험 환경을 고려하여 <표 114>와 같이 오픈스택 프로젝트를 구성하고, 하드웨어 시스템 구성에 따라 클라우드 환경을 구축함

- 각 서버 별 오픈스택 클라우드 환경을 위해 설치된 주요 소프트웨어 스택은 <표 115>, <표 116>와 같음

항목	프로젝트명	설명
· Compute	· Nova	· 가상머신 생성/삭제/메모리 관리 등의 라이프사이클 관리
· Networking	· Neutron	· 오픈스택 네트워크 관리
· Block Storage	· Cinder	· Nova 가상머신의 물리적인 디스크 공간을 제공
· Identity Service	· Keystone	· 오픈스택 서비스들의 사용 인증, 인가 서비스를 제공
· Image Service	· Glance	· 가상머신을 위한 OS 이미지 저장 및 관리 서비스를 제공

<표 114> 오픈스택 프로젝트 구성 리스트

패키지명	버전	설명
· Openstackit	· Queens	· 에뮬레이션 클라우드 환경 관리 및 제어를 위한 웹 포탈 솔루션
· Openstack	· Queens	· 에뮬레이션을 위한 클라우드 환경 제공 · 설치된 주요 모듈: Nova, Neutron, Glance, Cinder
· MariaDB	· 10.3.37	· 에뮬레이션 클라우드 환경 솔루션을 위한 데이터베이스
· Docker Engine	· 1.12.6	· 도커 컨테이너 배포 및 관리 · 주요 소프트웨어가 도커 컨테이너 기반으로 실행
· Openvswitch	· 2.9.0	· 가상 네트워크 관리 및 제어
· QEMU-KVM	· 2.11.1	· 가상 서버 에뮬레이터

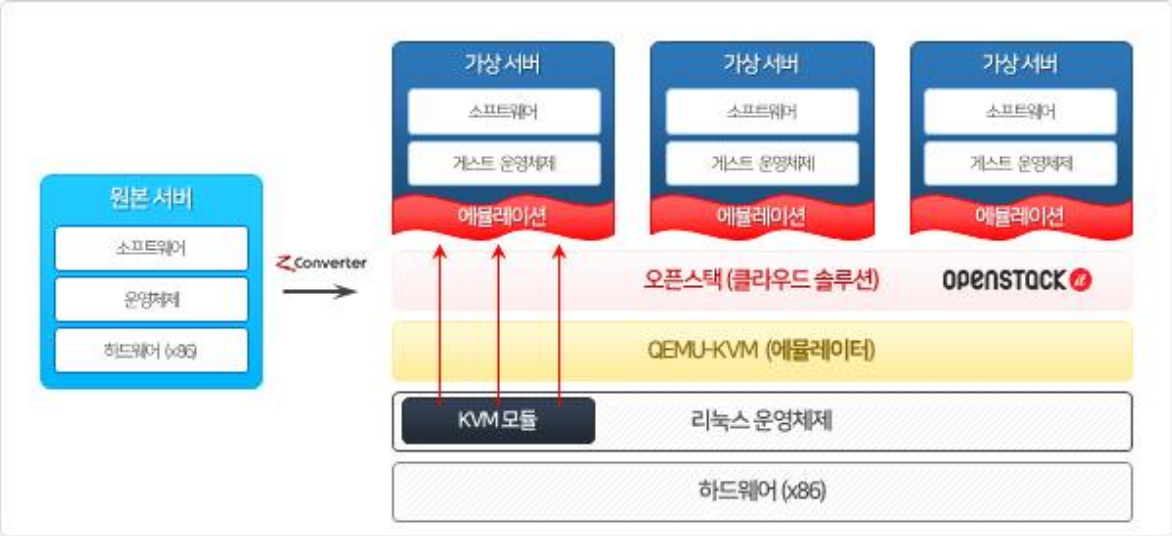
<표 115> 클라우드 환경 주요 소프트웨어 스택 (서버 1)

패키지명	버전	설명
· Openstack	· Queens	· 에뮬레이션을 위한 클라우드 환경 제공 · 설치된 주요 모듈: Nova, Neutron
· Docker Engine	· 1.12.6	· 도커 컨테이너 배포 및 관리 · 주요 소프트웨어가 도커 컨테이너 기반으로 실행
· Openvswitch	· 2.9.0	· 가상 네트워크 관리 및 제어
· QEMU-KVM	· 2.11.1	· 가상 서버 에뮬레이터

<표 116> 클라우드 환경 주요 소프트웨어 스택 (서버 2)

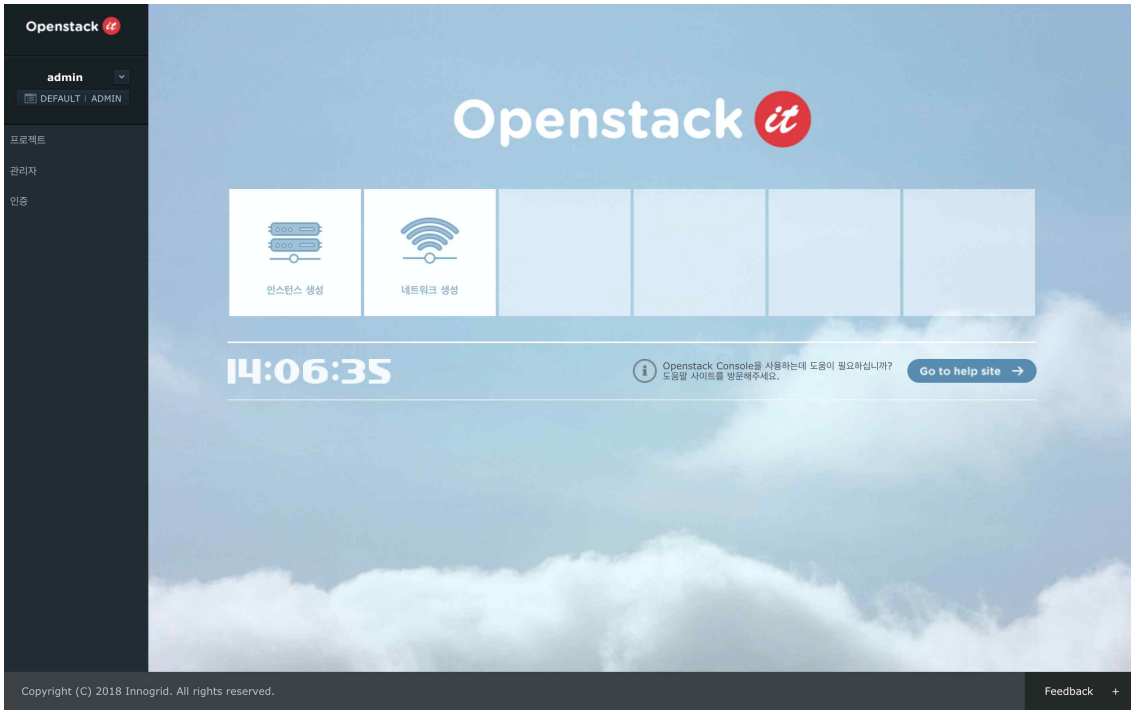
○ 오픈스택 기반 클라우드 인프라의 운영 및 관리를 위한 오픈스택잇(Openstackit) 설치

- 구축된 클라우드 환경을 운영하고 관리하기 위한 오픈스택잇 솔루션을 설치하여 <그림 106>와 같은 형태의 에뮬레이션 시험 환경을 제공함



<그림 106> 오픈스택잇 기반 에뮬레이션 시험 환경 구성도

- 클라우드 환경에서 에뮬레이션 시험 환경 제공을 위한 기능은 <그림 107>과 같이 오픈스택잇 사용자 화면을 통해 제공함



<그림 107> 오픈스택잇: 메인화면

<그림 108> 오픈스택잇: 클라우드 환경 내 모든 컴퓨팅 서버 정보 조회 화면

관리자 > 컴퓨트 > 사양

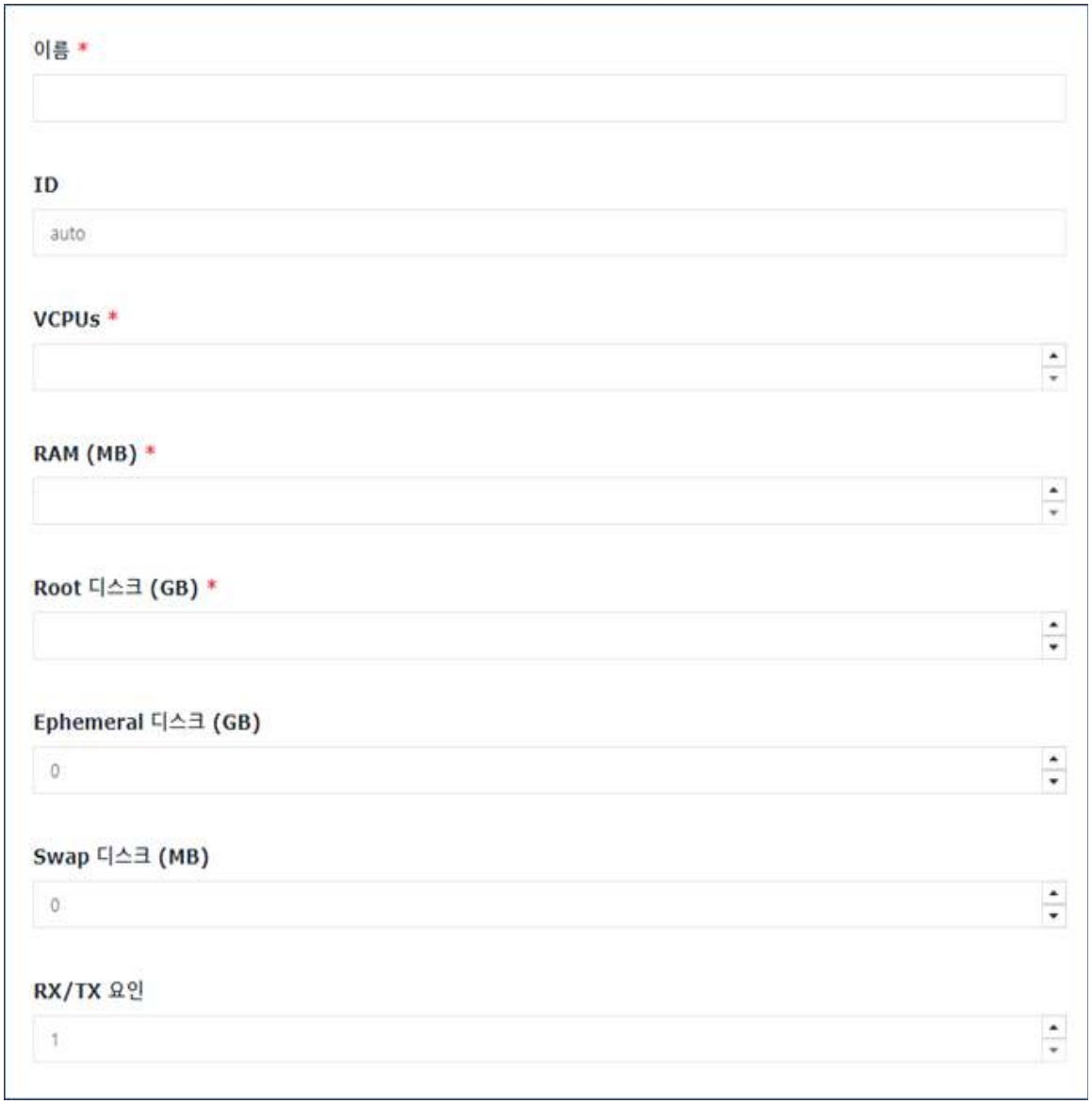
사양

8 항목 표시

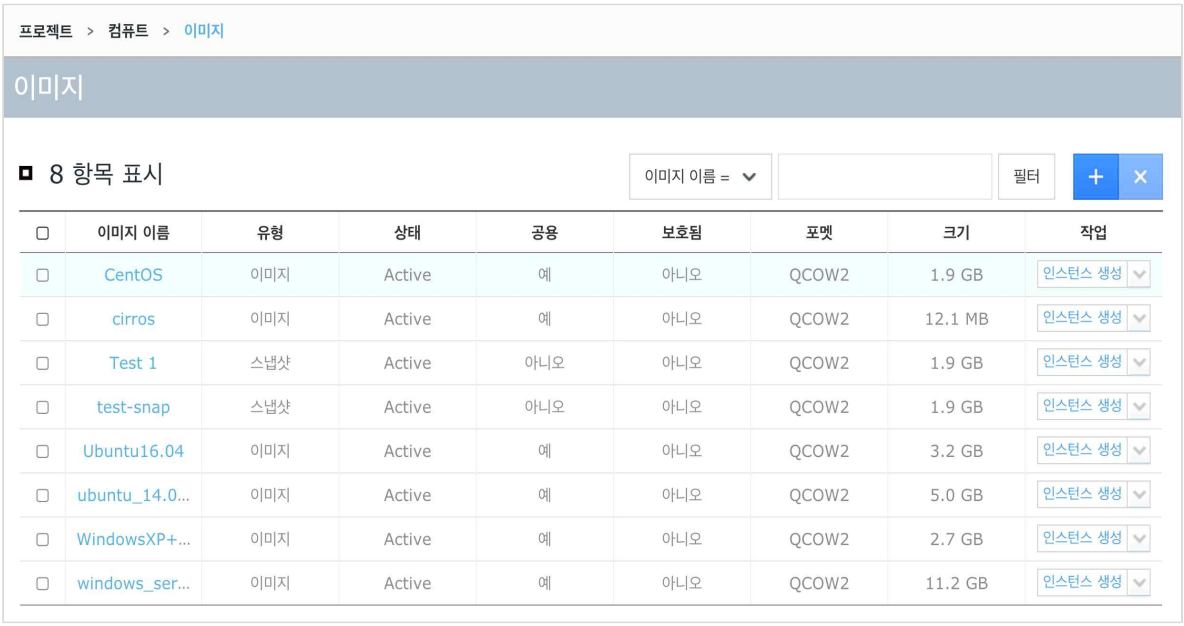
필터

☐	사양 이름	VCPUs	RAM	Root 디...	Ephem...	Swap 디...	RX/TX ...	ID	공용	메타데이터
☐	m1.large	4	8GB	80GB	0GB	0MB	1.0	4	예	아니오
☐	m1.medi...	2	4GB	40GB	0GB	0MB	1.0	3	예	아니오
☐	m1.small	1	2GB	20GB	0GB	0MB	1.0	2	예	아니오
☐	m1.tiny	1	512MB	1GB	0GB	0MB	1.0	1	예	아니오
☐	m1.xlarge	8	16GB	160GB	0GB	0MB	1.0	5	예	아니오
☐	t1.mediu...	2	4GB	100GB	0GB	0MB	1.0	9eeea6af...	예	아니오
☐	t1.small	1	2GB	100GB	0GB	0MB	1.0	311da8e...	예	아니오
☐	t2.mediu...	2	4GB	50GB	0GB	0MB	1.0	4e2d75c...	예	아니오

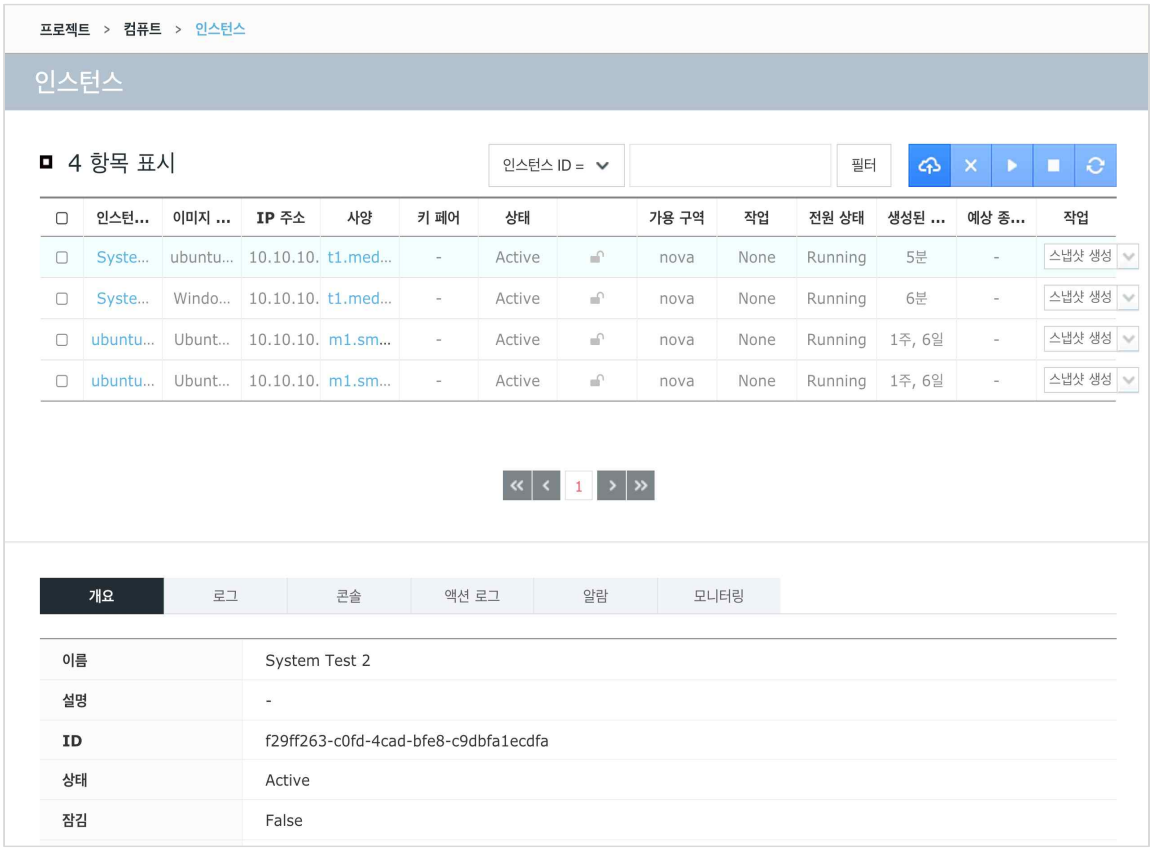
<그림 109> 오픈스택잇: 가상 서버의 사양 관리 화면



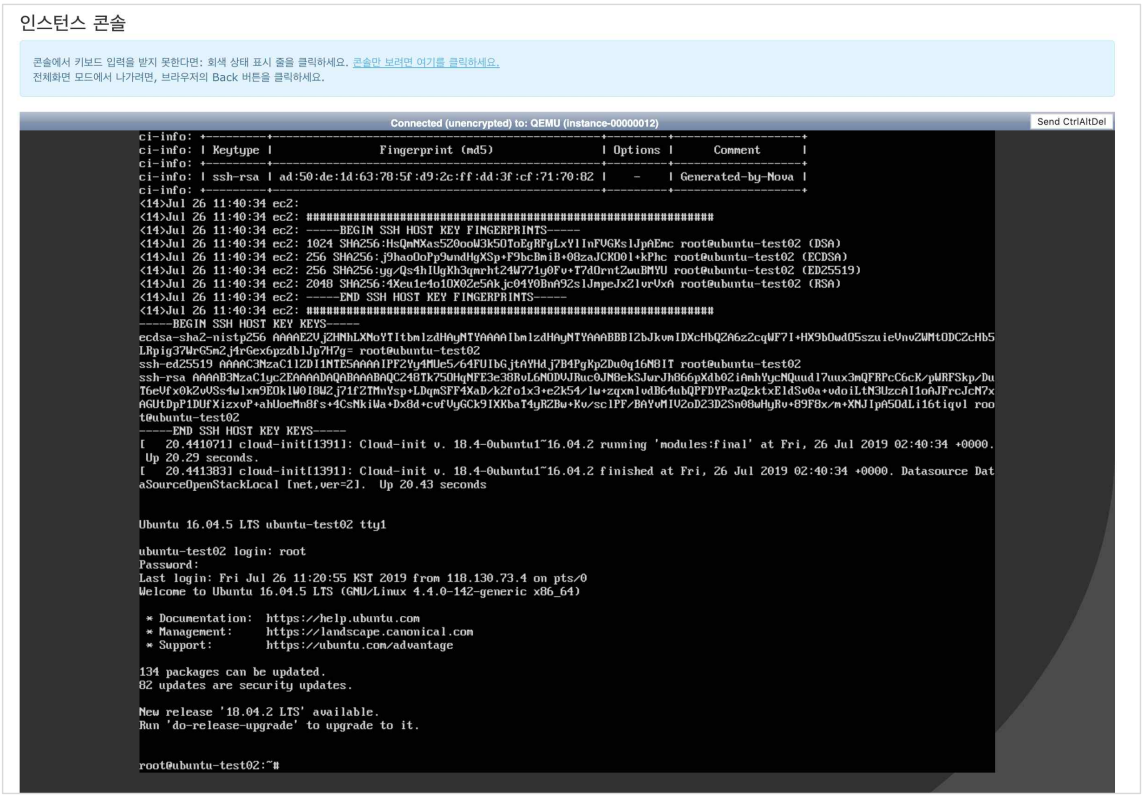
<그림 110> 오픈스택잇: 가상 서버 사양 생성 정보 입력 창



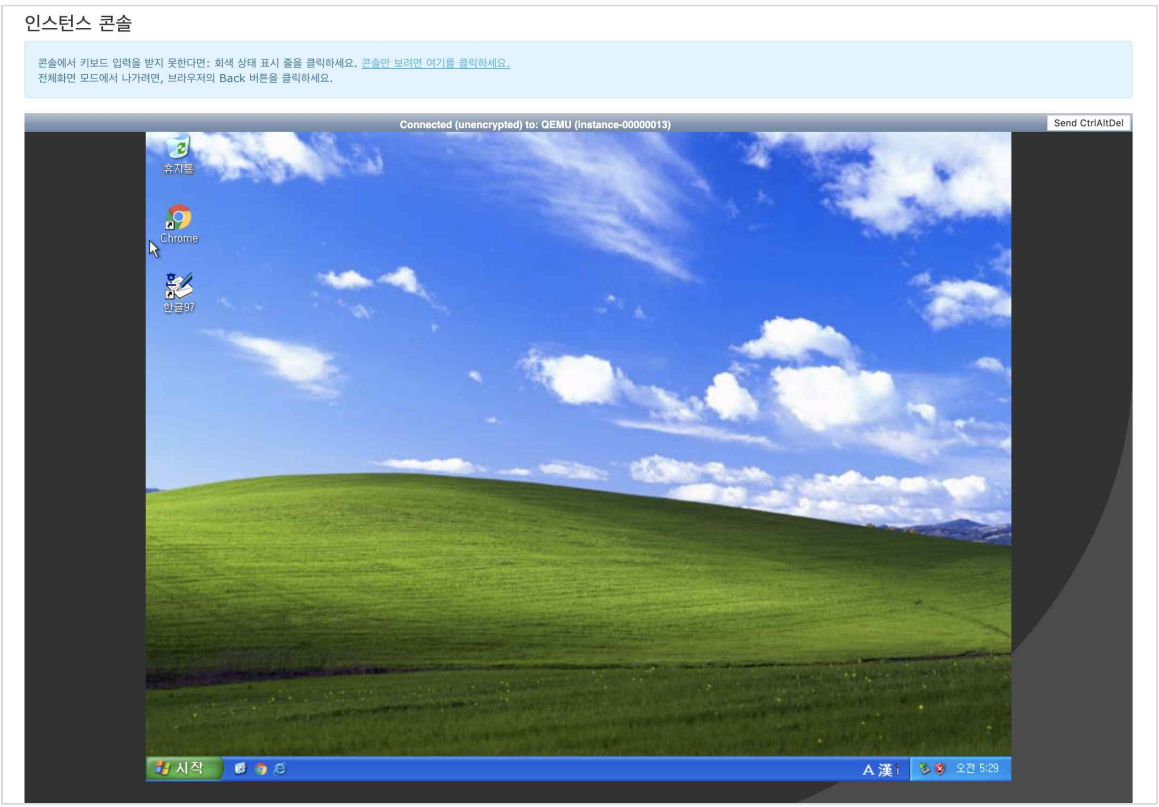
<그림 111> 오픈스택잇: 이미지 및 스냅샷 정보 조회 화면



<그림 112> 오픈스택잇: 가상 서버 정보 조회 화면



<그림 113> 오픈스택잇: 가상 서버 콘솔 접속 화면 1 (리눅스)



<그림 114> 오픈스택잇: 가상 서버 콘솔 접속 화면 2 (윈도우)

○ 에뮬레이션 시험을 위해 필요한 오픈스택잇 기능 항목 도출 및 검증

- 클라우드 환경에서 에뮬레이션 시험 환경 제공을 위해 필요한 주요 기능을 <표 117>와 같이 도출하고 검증함

구분	세부 기능 항목	결과	구분	세부 기능 항목	결과
이미지	이미지 생성	정상	볼륨	볼륨 생성	정상
	이미지 삭제	정상		볼륨 삭제	정상
	이미지 조회	정상		볼륨 조회	정상
	이미지 → 가상 서버 생성	정상		볼륨 연결	정상
스냅샷	스냅샷 생성	정상	네트워크	네트워크 생성	정상
	스냅샷 삭제	정상		네트워크 삭제	정상
	스냅샷 조회	정상		네트워크 조회	정상
	스냅샷 → 이미지 변경	정상		라우터 생성	정상
	스냅샷 → 가상 서버 생성	정상		라우터 삭제	정상
가상서버	가상 서버 사양 관리	정상		라우터 조회	정상
	가상 서버 생성	정상		보안그룹 생성	정상
	가상 서버 삭제	정상		보안그룹 삭제	정상
	가상 서버 시작/종료/재부팅	정상		보안그룹 조회	정상
	가상 서버 콘솔 연결	정상		유동 IP 추가	정상
	가상 서버 네트워크 통신	정상		유동 IP → 가상 서버 연결	정상

<표 117> 에뮬레이션 시험 환경 제공을 위한 오픈스택잇 주요 기능 검증

3. 데이터세트 유형 전자기록의 에뮬레이션 시험

- 데이터 세트 유형 전자기록의 에뮬레이션 시험에 앞서 에뮬레이션 시험 대상 시스템 선정 준비를 진행함
- 선정된 에뮬레이션 시험 대상 시스템의 에뮬레이션 전환 시험을 테스트함
- 에뮬레이션 시험 대상 시스템을 선정하고, 선정된 시스템별 에뮬레이션 시험을 통해 원본 서버와 동일한 모습으로 재현이 되는지 검증 과정을 진행함
- 에뮬레이션 시험 및 검증 과정을 거쳐 에뮬레이션 절차와 정합검 검증 항목들을 도출함

3.1 에뮬레이션 시험 대상 시스템 선정 준비

- 에뮬레이션 시험 대상 시스템 선정을 위해 에뮬레이션 시험에서 사용할 용어를 정의하고, 시스템 유형을 시스템 노드와 구성요소별로 정의함
- 에뮬레이션 대싱 시스템 및 시스템 노드에 대한 세부 요구 정보 항목을 정의하고, 정의된 유형 항목을 기반으로 시스템 정보 입력 양식을 도출함

<그림 115> 에뮬레이션 시험 대상 시스템 선정 과정

○ 용어 정의

- 에뮬레이션 시험 대상 시스템 선정을 위한 용어를 정의함
- 에뮬레이션 시험 대상 시스템은 '시스템' 혹은 '에뮬레이션 시스템'으로 정의하며, 시스템은 1개의 '시스템 운영체제'와 1개 이상의 '시스템 구성요소'로 구성된 1대 이상의 '시스템 노드'로 구성됨
- '시스템 노드'는 시스템을 구성하는 하나의 서버(일반적으로 말하는 PC 혹은 컴퓨터), '시스템 운영체제'는 시스템 노드에서 시스템 구성요소를 실행하는데 요구되는 운영체제, '시스템 구성요소'는 하나의 시스템이 실행되기 위해 필요한 핵심 소프트웨어로 <표 118>과 같이 정의함

항목	설명
· 시스템 노드	· 시스템을 구성하는 하나의 서버(일반적으로 말하는 PC, 컴퓨터)
· 시스템 구성요소	· 하나의 시스템이 실행되기 위해 필요한 핵심 소프트웨어 (DB, WAS 등등)
· 시스템 운영체제	· 시스템 노드에서 시스템 구성 요소를 실행하는데 요구되는 운영체제(OS)

<표 118> 시스템 구성 항목 용어 정의

- 또한 '시스템 구성요소'를 '애플리케이션', '데이터베이스', '기타 서비스'로 분류하여 <표 119>과 같이 정의함

항목	설명
· 애플리케이션	· WAS(Web Application Server)와 같이 시스템의 서비스를 제공하는 소프트웨어
· 데이터베이스	· 애플리케이션 구동에 필요한 데이터베이스 · 예: MySQL, MSSQL 등
· 기타 서비스	· 애플리케이션 구동에 필요한 3rd-Party 프로그램 · 예: Apache2, Nginx, 외부 파일 서버 등

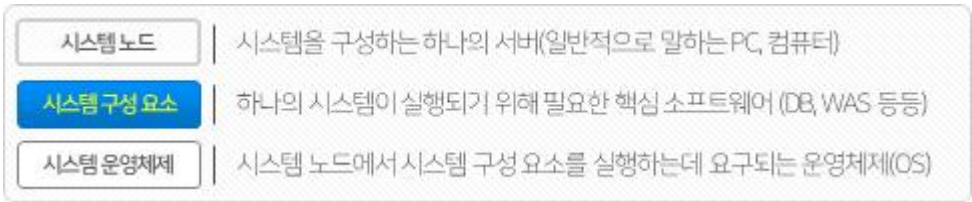
<표 119> 시스템 구성요소 세부 항목 용어 정의

○ 시스템 유형 정의

- 시스템 유형 정의는 에뮬레이션 시험 대상 시스템의 유형을 정의함
- <표 119>, <그림 118>와 같이 '시스템'을 '시스템 노드', '시스템 구성 요소' '시스템 운영체제'로 구성하여 크게 3가지 유형으로 시스템을 구분함
- 시스템 유형 1은 1개의 시스템노드에 모든 구성 요소들이 구성되어 있는 시스템으로, 시스템 유형 2는 각 구성요소들이 단일 노드별로 구분되어 구성되어 있는 시스템으로, 시스템 유형 3은 시스템 유형 2에서 단일 노드가 아닌 다수의 노드로 구성되어있는 시스템으로 정의하며 <그림 117>, <그림 118> <그림 119>과 같음

유형	시스템 노드	시스템 구성요소
· 유형 1	· 1대의 시스템 노드로 구성	· 노드1: 애플리케이션 1개, 데이터베이스 1개, 기타서비스 0~1개
· 유형 2	· 2~3대의 시스템 노드로 구성	· 노드1: 애플리케이션 1개, 기타 서비스 0~1개 · 노드2: 데이터베이스 1개 · 노드3: 기타서비스 0~1개
· 유형 3	· 시스템 유형 2와 같은 구성이지만 4대 이상의 시스템 노드로 구성	· 노드N: 애플리케이션, 데이터베이스, 기타 서비스 N개

<표 120> 시스템 유형 상세



<그림 116> 시스템 유형 구성

<그림 117> 시스템 유형 1

<그림 118> 시스템 유형 2

<그림 119> 시스템 유형 3

○ 시스템 세부 항목 정의

- 시스템 세부 항목 정의는 에뮬레이션 대상 시스템 및 시스템 노드에 대한 요구 정보 항목을 정의함

- 앞서 정의한 시스템 용어 및 유형에 따라 시험 대상 선정을 위해 필요한 항목들에 대해 시스템 항목은 <표 121>, 시스템을 구성하는 시스템 노드 항목은 <표 122>과 같이 정의함

항목	설명
· 시스템명	· 시스템을 구분하기 위한 명칭
· 시스템 설명	· 해당 시스템이 어떠한 서비스를 제공하는지 등의 간단한 설명
· 시스템 유형	· 앞에서 정의된 시스템 유형 中 택 1
· 시스템 상세 구성도	· 시스템 노드와 구성 요소에 대한 상세 항목 및 각 요소 간의 연결이 어떻게 구성되어 있는지 표현된 구성도 제시
· 시스템 구성요소 설정 항목	· IP 주소와 같이 시스템이 다른 환경으로 재현될 때 변경되는 값들에 대한 시스템 구성 요소들 간에 설정되어야 할 항목에 대한 정보 제시
· 기타 요구 사항	· 위에서 제시된 항목 외에 해당 시스템에서 반드시 고려해야 하거나 필요한 항목들이 있으면 제시

<표 121> 에뮬레이션 대상 시스템 요구 항목

항목	설명
· 시스템 노드명	· 시스템 노드를 구분하기 위한 명칭
· 시스템 구성요소 (소프트웨어)	· 해당 시스템 노드에서 설치되고 실행될 소프트웨어에 대한 상세 정보
· 운영체제 (버전포함)	· 운영되는 서버의 운영체제 종류 및 버전에 대한 상세 정보
· 최소/권장 CPU 코어 수	· 소프트웨어가 실행되기 위한 최소 또는 권장 CPU 코어 수
· 최소/권장 메모리 크기 (GB)	· 소프트웨어가 실행되기 위한 최소 또는 권장 메모리 크기
· 필요 디스크 크기 (GB)	· 운영체제가 설치되는 기본 루트 디스크의 크기 · 일반적으로 리눅스에서는 /, 윈도우 C:\ 경로의 크기를 말함
· 기타 추가 디스크 크기 (GB)	· 기본 디스크 사이즈 외에 추가로 필요한 디스크의 크기 · 일반적으로 윈도우의 D:\ 경로와 같이 추가적으로 사용되는 디스크의 크기를 말함
· 기타 요구 사항	· 위에서 제시된 항목 외에 해당 시스템 노드에서 반드시 고려해야 하거나 필요한 항목들이 있으면 제시

<표 122> 에뮬레이션 대상 시스템 노드 요구 항목

- 정의된 유형 항목을 기반으로 하여 시스템 정보 입력 양식을 <그림 120>와 같이 정의하고, <그림 121>와 같이 에뮬레이션 시험 대상 시스템을 선정토록 함

<그림 120> 에뮬레이션 대상 시스템 요구 정보 양식

시스템 노드명	명칭
시스템 구성요소	• 구성요소 • 구성요소
운영체제	운영체제
최소/권장 CPU 코어 수	코어 수
최소/권장 메모리 크기	메모리 크기
필요 디스크 크기	디스크 크기
기타 추가 디스크 크기	• 추가 디스크 1 크기 • 추가 디스크 2 크기
기타 요구 사항	• 항목: 설명 • 항목: 설명

<그림 121> 에뮬레이션 대상 시스템 노드 요구 정보 양식

○ 기록관리평가시스템 예시

- 시스템 구성요소 설정항목은 IP주소와 같이 시스템이 다른 환경으로 재현될 때 변경되는 값들에 대한 시스템 구성 요소들 간에 설정되어야 할 항목에 대한 정보를 작성함

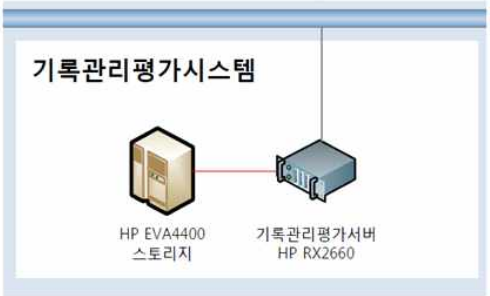
- 기록관리평가시스템의 경우 에뮬레이션 시험과정에서 아래 <표 123>와 같이 변경되었음

항목	원본시스템	대상시스템
· IP 주소	· 192.168.100.150	· 10.10.10.15
· 호스트 파일 내 IP 정보	· localhost 192.168.100.150	· localhost 10.10.10.15
· tomcat 설정	· 변동사항 없음	· 변동사항 없음
· 소스수정	· 변동사항 없음	· 변동사항 없음

<표 123> 시스템 구성요소 설정 항목 예시

시스템명	기록관리평가시스템
시스템 설명	-기록관리평가 대상기관 정보, 평가지표 등록 및 운영관리 - 프로그램 변경 및 운영지원
시스템 유형	
시스템 구성요소 설정 항목	• Ip설정: OS IP설정 파일 수정 • Host 파일: hosts 파일내 ip 정보 수정 • tomcat 설정 : tomcat 프로그램 설정 파일 수정 • 소스 수정 : Web, WAS 소스내 IP 정보 수정 • 서버 코어 추가시 orade 라이선스 변경 필요
기타 요구사항	• 항목: 설명 • 항목: 설명

시스템 상세 구성도



<그림 122> 양식에 따라 작성된 시스템 예시 (기록관리평가시스템)

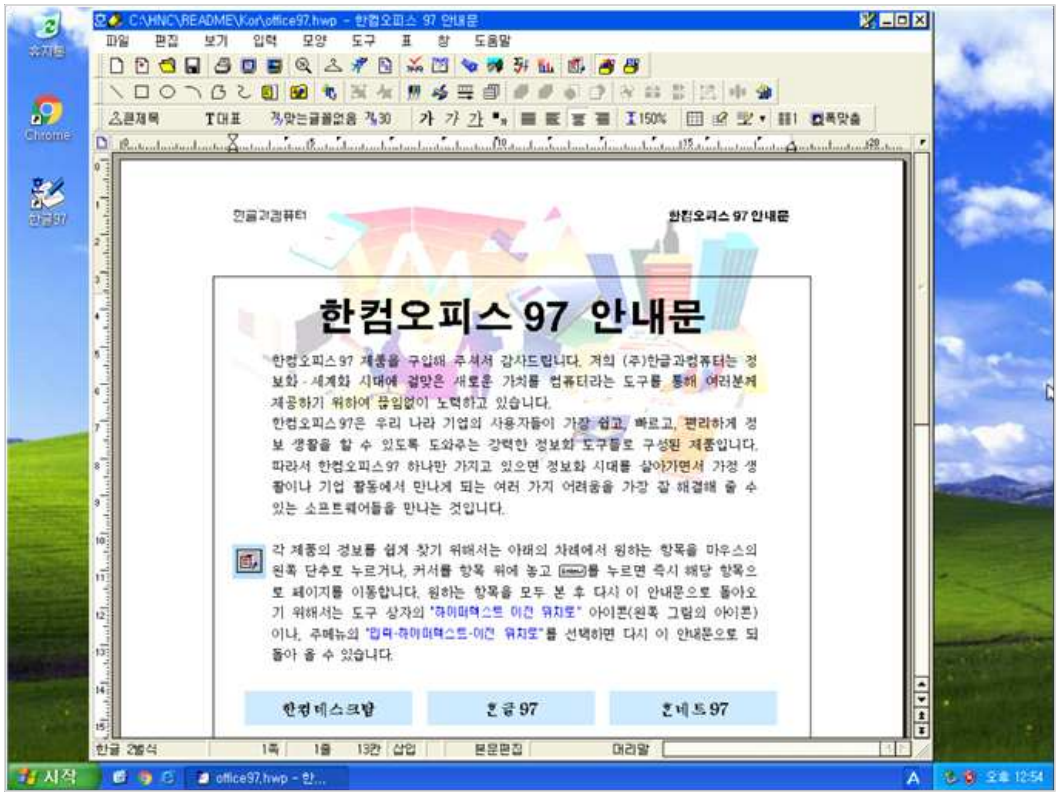
3.2 에뮬레이션 시험 대상 시스템 사전 테스트

- 에뮬레이션 시험 대상 선정하기 전에 윈도우 XP 운영체제와 한글 97, 우분투 운영체제와 MySQL이 구동되는 임의의 테스트 시스템을 생성하여 모의 테스트를 진행함

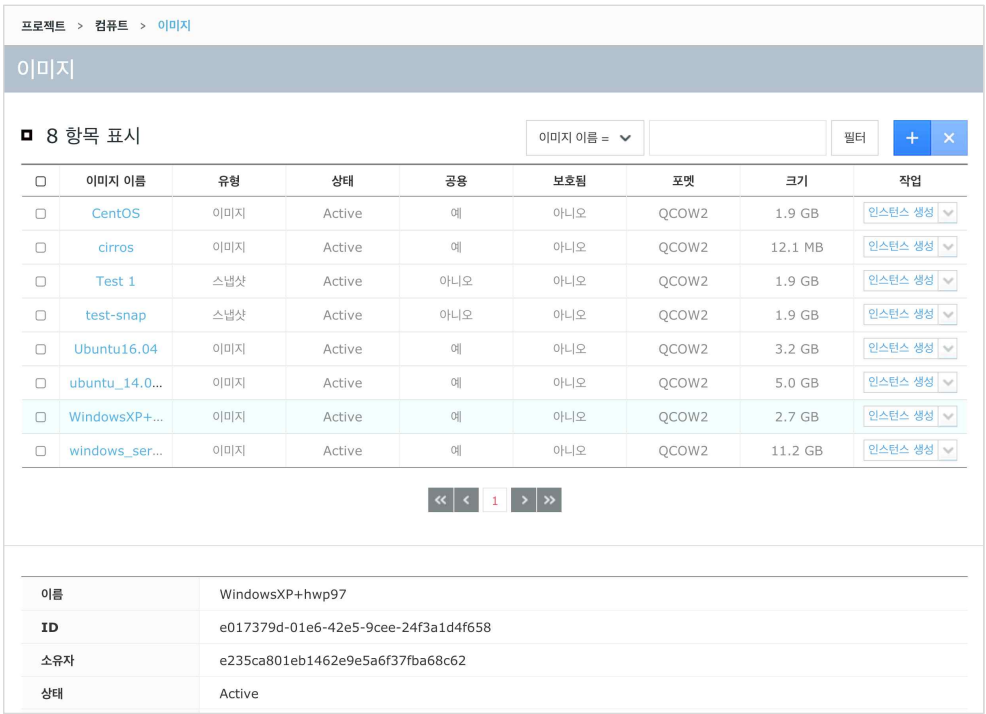
○ 항목 검증을 위한 테스트 진행

- 국가기록원의 실 운영시스템을 시험 대상으로 선정하기 전 에뮬레이션 환경에서 시스템 유형1에 해당하는 임의의 테스트 시스템을 생성하여 모의 테스트를 진행함

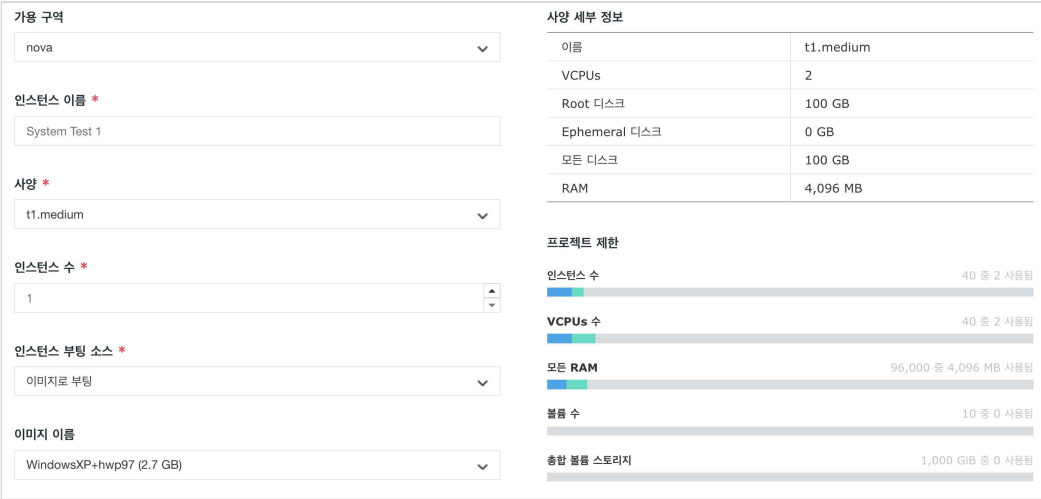
- <그림 123>는 윈도우 XP 운영체제에 한글 97이 구동되는 시스템을(테스트 시스템 1), <그림 124>은 우분투 14.04 운영체제에 MySQL 5.5가 구동되는 시스템(테스트 시스템 2)을 재현함



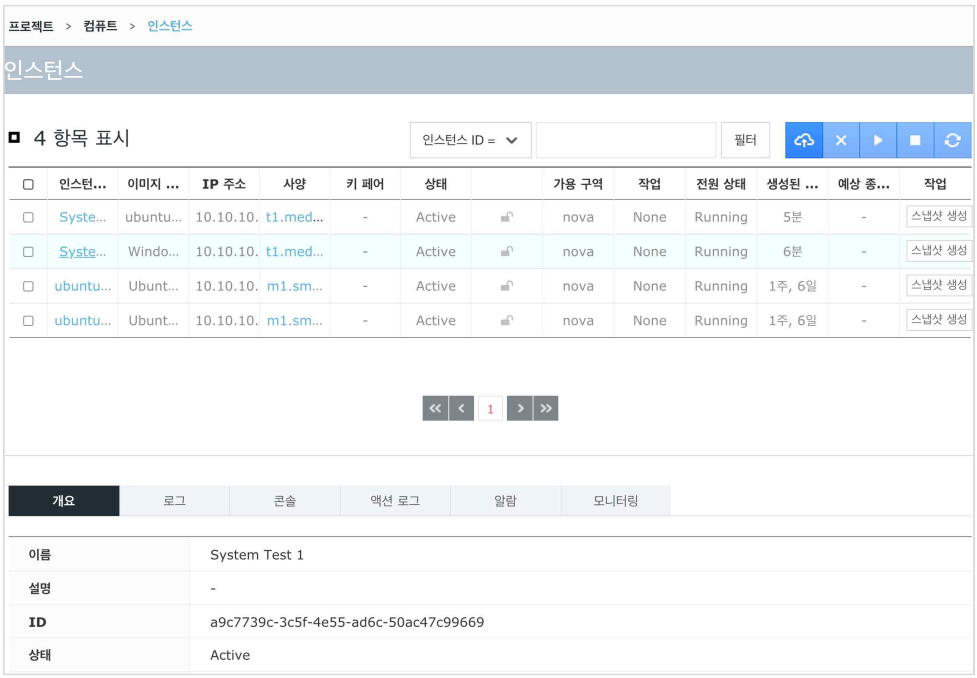
<그림 123> 테스트시스템1: 원본 데스크톱 화면



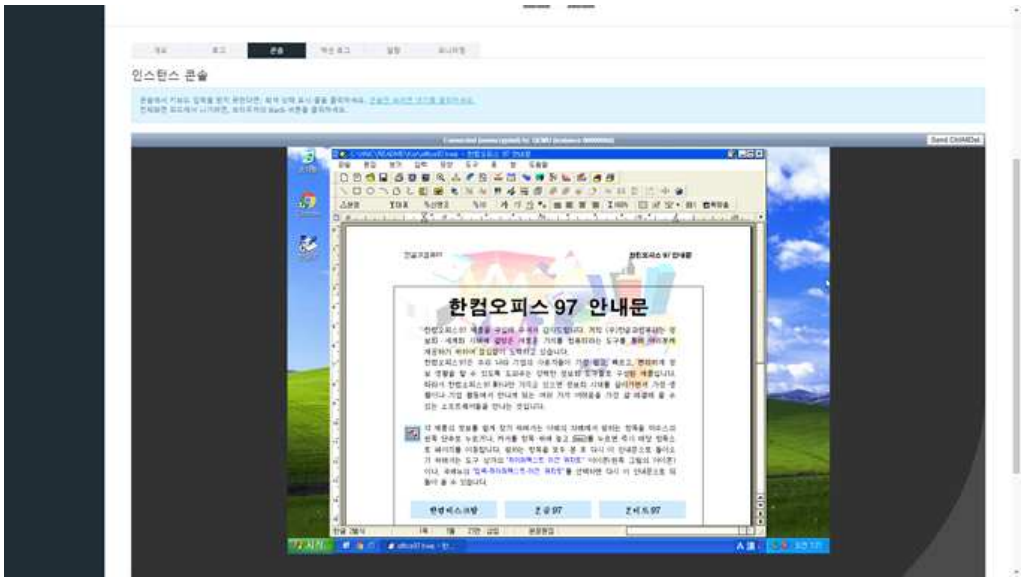
<그림 124> 테스트시스템1: 클라우드 환경에 업로드된 시스템 노드 이미지 정보 조회 화면



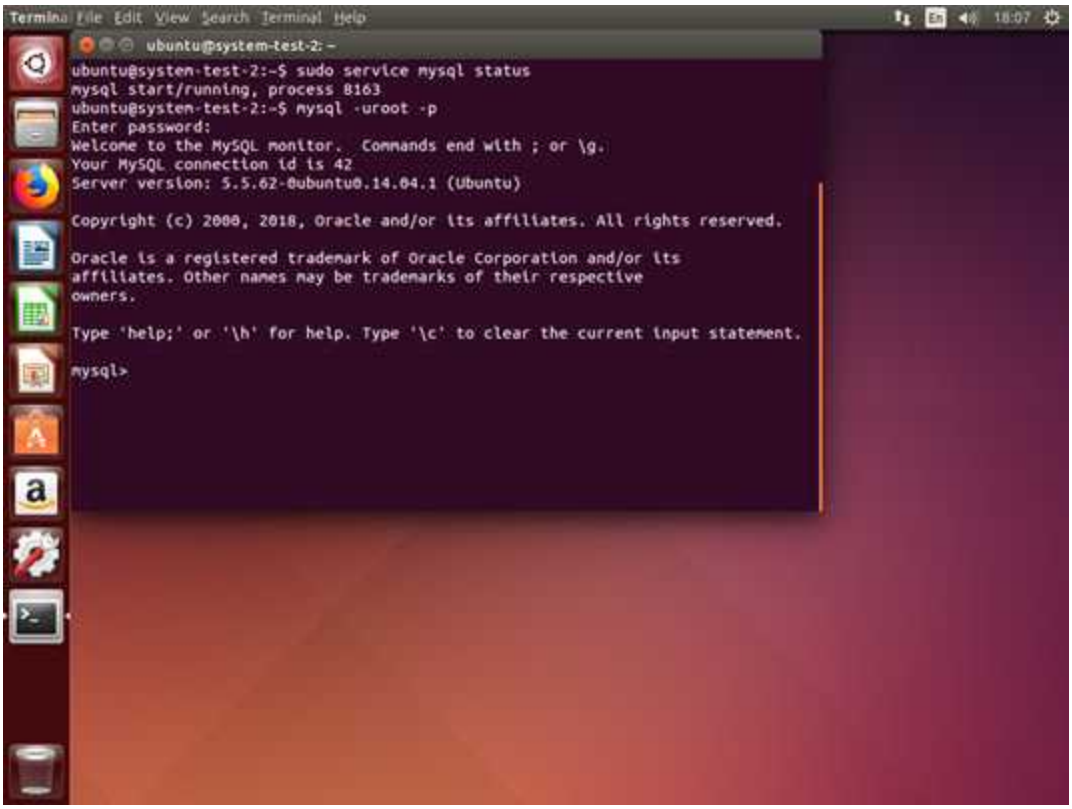
<그림 125> 테스트시스템1: 시스템 노드 생성 화면



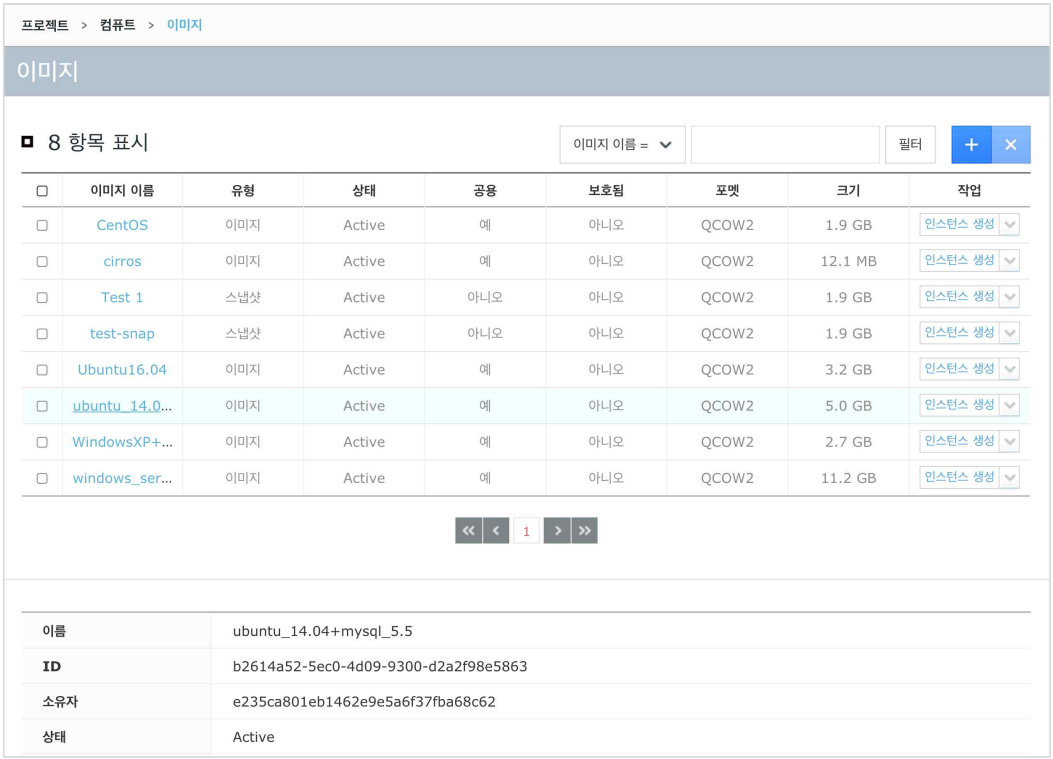
<그림 126> 테스트시스템1: 생성된 시스템 노드 정보 조회 화면



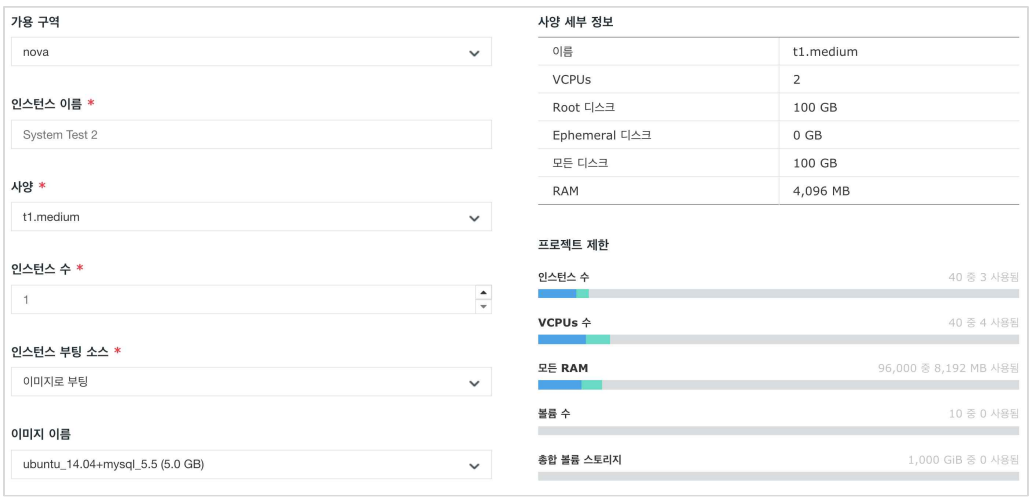
<그림 127> 테스트시스템1: 재현된 데스크톱 화면



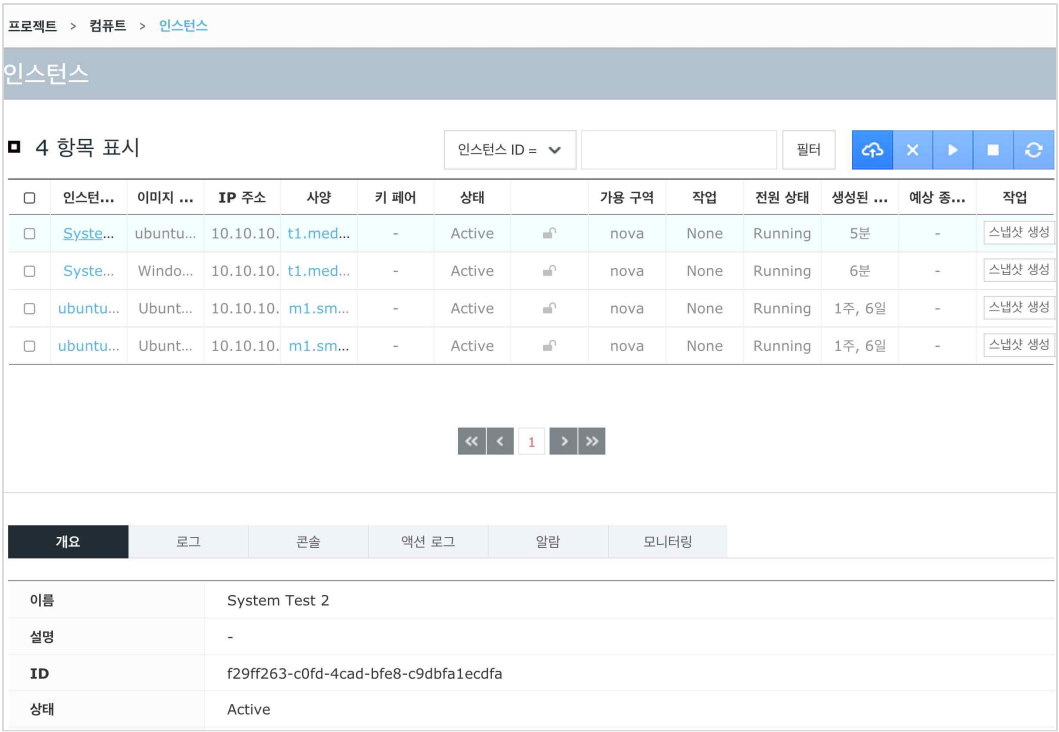
<그림 128> 테스트시스템2: 원본 데스크톱 화면



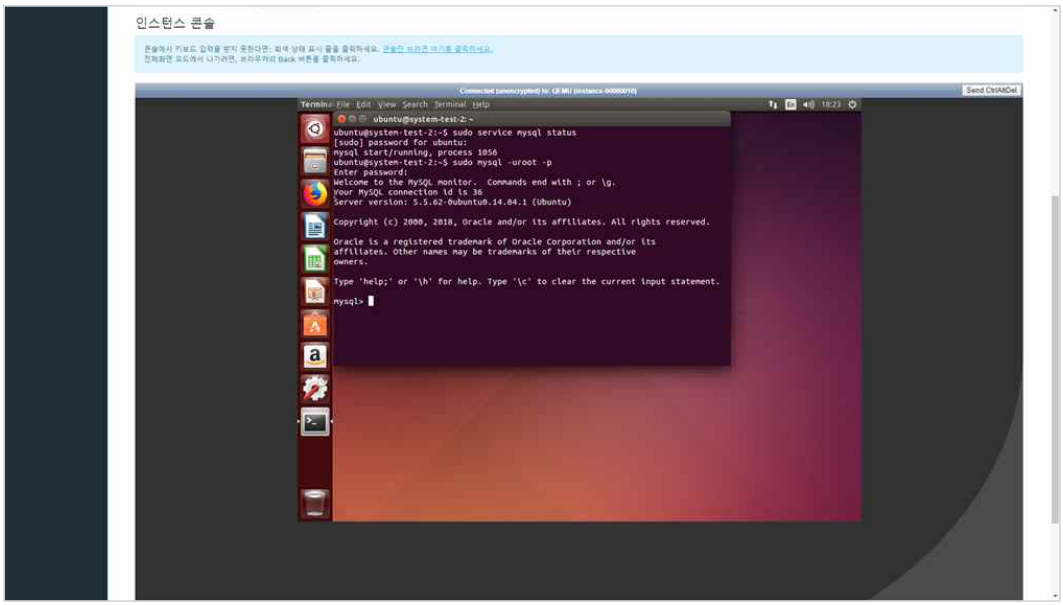
<그림 129> 테스트시스템2: 클라우드 환경에 업로드된 시스템 노드 이미지 정보 조회 화면



<그림 130> 테스트시스템2: 시스템 노드 생성 화면



<그림 131> 테스트시스템2: 생성된 시스템 노드 정보 조회 화면



<그림 132> 테스트시스템2: 재현된 데스크톱 화면

3.3 에뮬레이션 시험 대상 시스템 선정

▪ 시험 대상 선정을 위해 정의된 항목에 따라 세가지 에뮬레이션 시험 대상 시스템을 선정함

○ 에뮬레이션 시험 대상 시스템 선정

- 시험 대상 선정을 위해 정의된 항목에 따라 '기록 관리 교육 훈련 시스템', '가상 국세청 홈택스 시스템', 'MS-DOS 보석글' 세 가지의 에뮬레이션 시험 대상 시스템을 선정함
- '기록 관리 교육 훈련 시스템'은 기록 관리 교육 과정의 체계적인 관리를 위해 교육과정의 운영전반을 온라인으로 처리하는 시스템으로 WEB/WAS를 위한 서버와 DB를 위한 서버로 구성됨
- '가상 국세청 홈택스 시스템'은 국세청 홈택스 웹 서비스와 유사한 환경으로 구성된 시스템으로 WEB/WAS를 위한 서버와 DB를 위한 서버로 구성됨
- 'MS-DOS 보석글'은 MS-DOS 환경에서 보석글 프로그램을 구동할 수 있도록 구성한 시스템으로 MS-DOS를 위한 서버 하나로 구성됨

시스템명	설명	구성시스템노드
기록관리 교육훈련 시스템	· 기록 관리 교육 과정의 체계적인 관리를 위해 교육과정의 운영전반을 온라인으로 처리하는 시스템	· 기록관리교육 WEB서버 · 기록관리교육 DB서버
가상 국세청 홈택스 시스템	· 국세청 홈택스 웹 서비스와 유사한 환경으로 구성된 시스템	· 가상국세청홈택스 WEB서버 · 가상국세청홈택스 DB서버
MS-DOS 보석글	· MS-DOS에서 보석글 프로그램을 구동할 수 있도록 구성된 시스템	· MS-DOS 서버

<표 124> 에뮬레이션 시험 대상 시스템

3.4 선정된 시스템 별 에뮬레이션 시험 및 검증

- 선정된 시스템의 제약사항으로 인하여 유닉스 운영체제가 아닌 리눅스 운영체제로 대체하여 1차 에뮬레이션 시험을 진행함
- 실제 운영중인 시스템 제공의 문제점을 고려하여 가상의 국세청 홈텍스 웹 서비스를 개발하여 2차 에뮬레이션 시험을 진행함
- 구 전자문서 프로그램을 에뮬레이션 시험으로 재현하기 위해 MS-DOS 운영체제와 보석글 프로그램이 구동되는 환경을 선정하여 3차 에뮬레이션 시험을 진행함

3.4.1 1차 시험 및 검증: 기록 관리 교육 훈련 시스템

○ 1차 시험 및 검증은 '기록 관리 교육 훈련 시스템'으로 선정하여 진행하려 하였으나, 시스템의 실 환경 제공이 불가능하고, DB 서버에서 사용 중인 운영체제는 본 연구에서 진행하는 에뮬레이션 시험 환경에서는 지원되지 않은 운영체제임

- '기록관리교육DB서버'의 운영체제인 HP-UX는 휴렛 패커드(Hewlett Packard)에서 나온 유닉스 계열 운영체제로 휴렛 패커드에서 만든 전용 플랫폼 (PA-RISC, IA-64)에서만 동작함

- 본 연구를 위한 에뮬레이션 시험 환경은 x86 기반의 리눅스/윈도우 외에 다른 시스템은 지원하지 않음

○ 이러한 제약사항으로 1차 시험 및 검증 시스템은 '기록 관리 교육 훈련 시스템'과 유사한 환경으로 가상의 시스템을 <표 125>, <표 126>과 같이 구성하여 진행함

- 시험 노드2에서 DB서버의 운영체제는 원본 시스템의 HP-UX 대신에 리눅스 환경인 CentOS로 구성하여 진행함

시스템	· 기록 관리 교육 훈련 시스템
서버	· 기록관리교육 WEB서버
운영체제	· Windows Server 2012 Standard R2 (x64)
구성요소	· WebtoB v4.1 · JEUS v6.0

<표 125> 1차 시험 및 검증 시스템 시험 노드 1 구성 정보

시스템	· 기록 관리 교육 훈련 시스템
서버	· 기록관리교육 DB서버
운영체제	· CentOS 7.6 1810
구성요소	· Oracle 12g

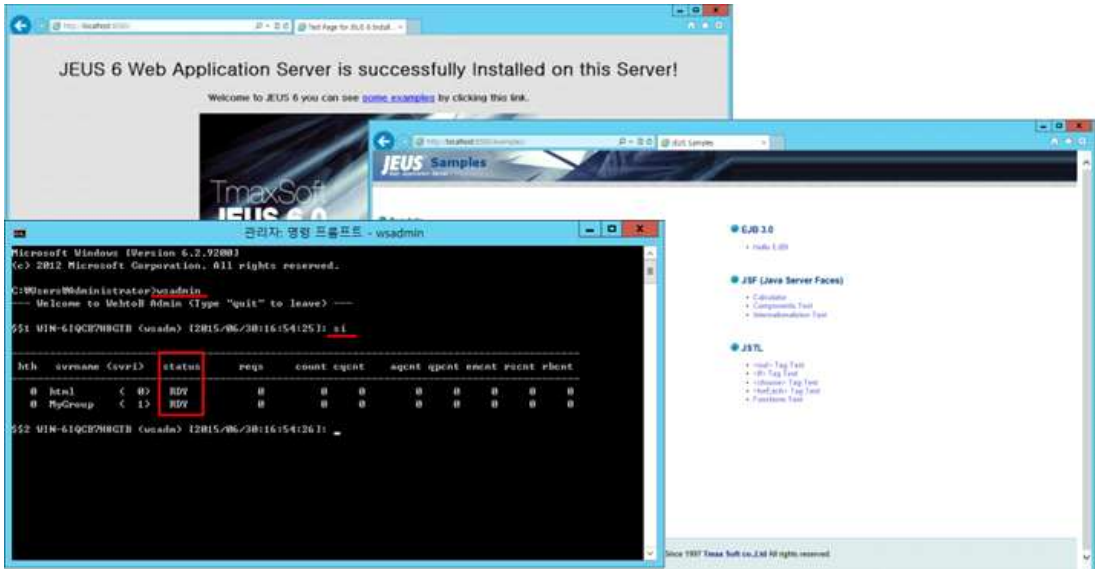
<표 126> 1차 시험 및 검증 시스템 시험 노드 2 구성 정보

○ 1차 시험 및 검증에서 '시험 노드 1'은 zConverter를, '시험 노드2'는 virt-p2v를 사용하여 각 물리 서버를 에뮬레이션을 위한 포맷의 가상 서버로 변환함

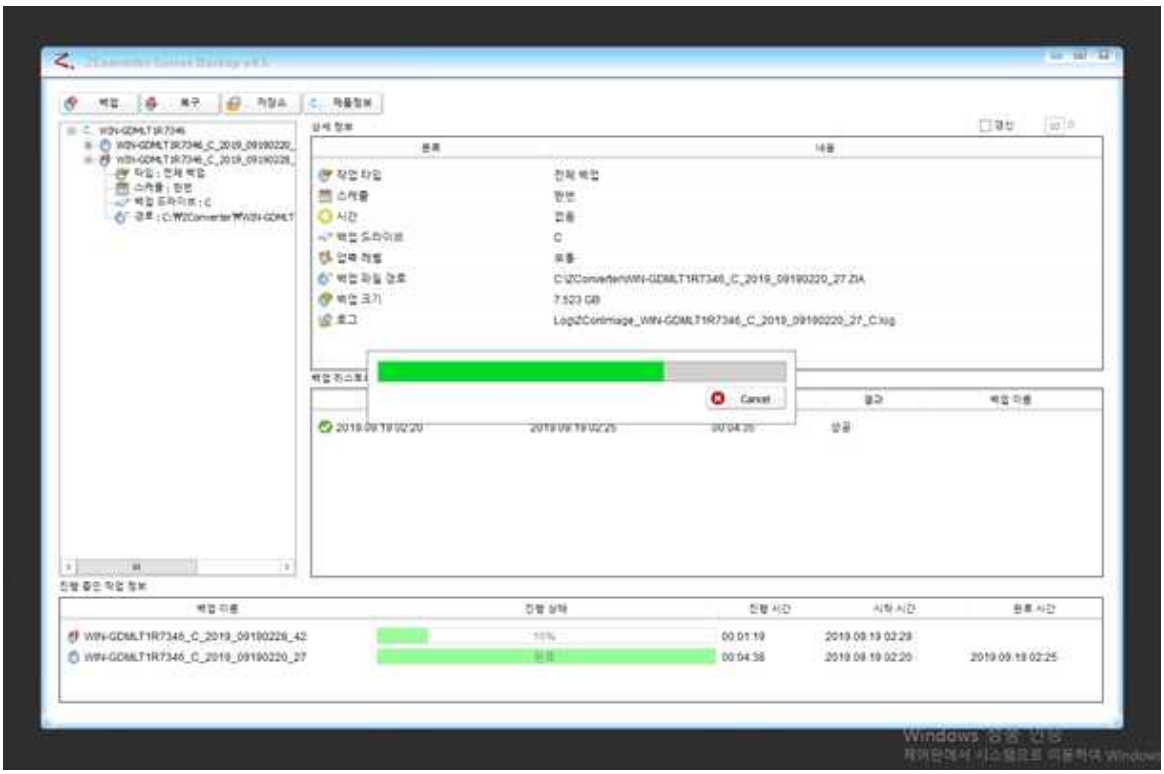
<그림 133> zConverter를 이용한 변환

<그림 134> virt-p2v를 이용한 변환

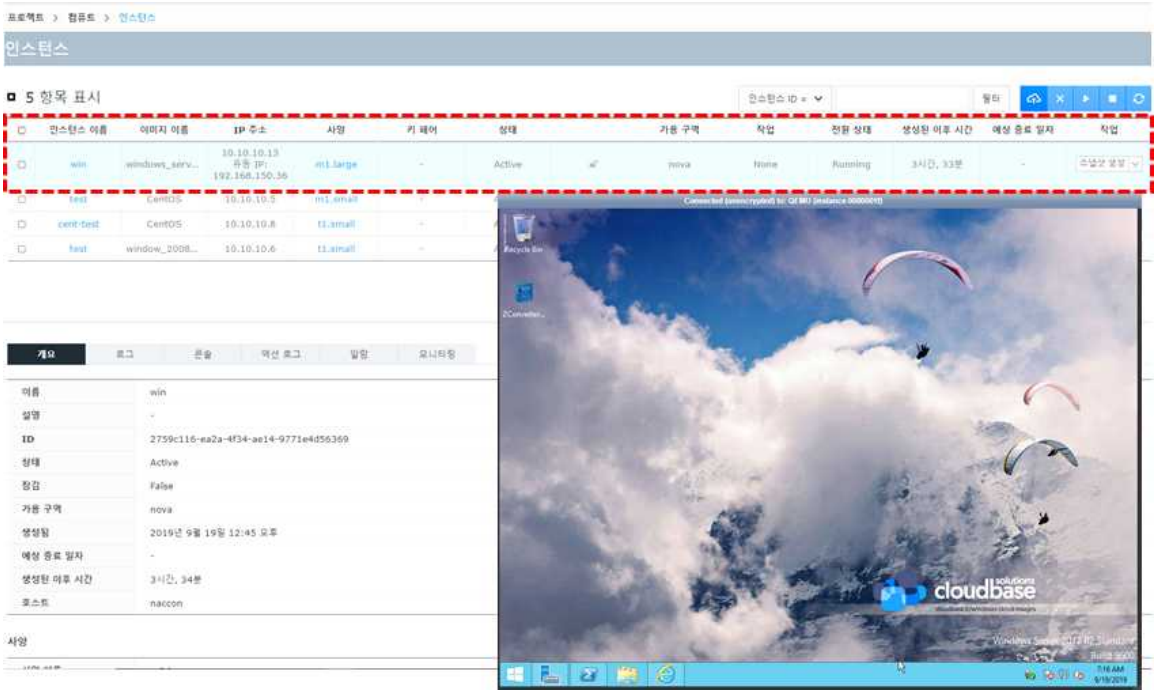
○ 시험 노드 1 (기록관리교육 WEB서버)의 에뮬레이션 시험 및 검증 결과는 아래 <그림 135 ~ 145>와 같음



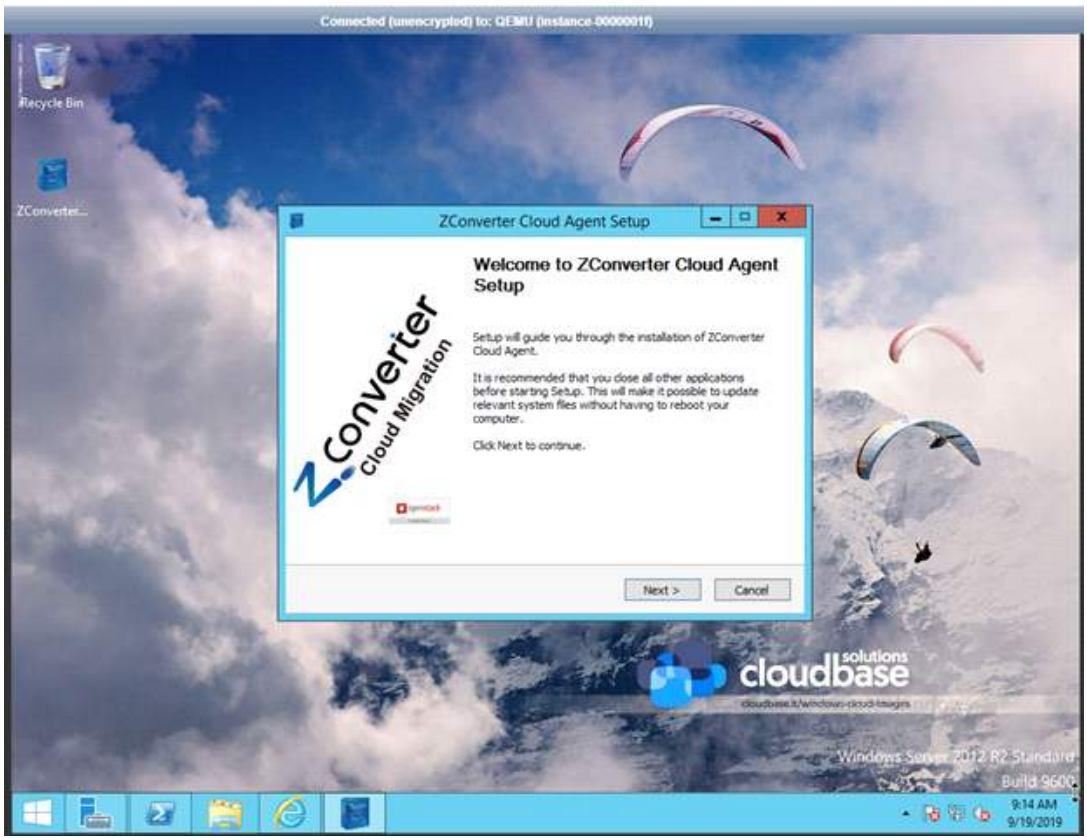
<그림 135> 시험 노드 1 환경 구성 (WebtoB & JEUS)



<그림 136> zConverter를 이용한 시험 노드 1의 백업 이미지 생성



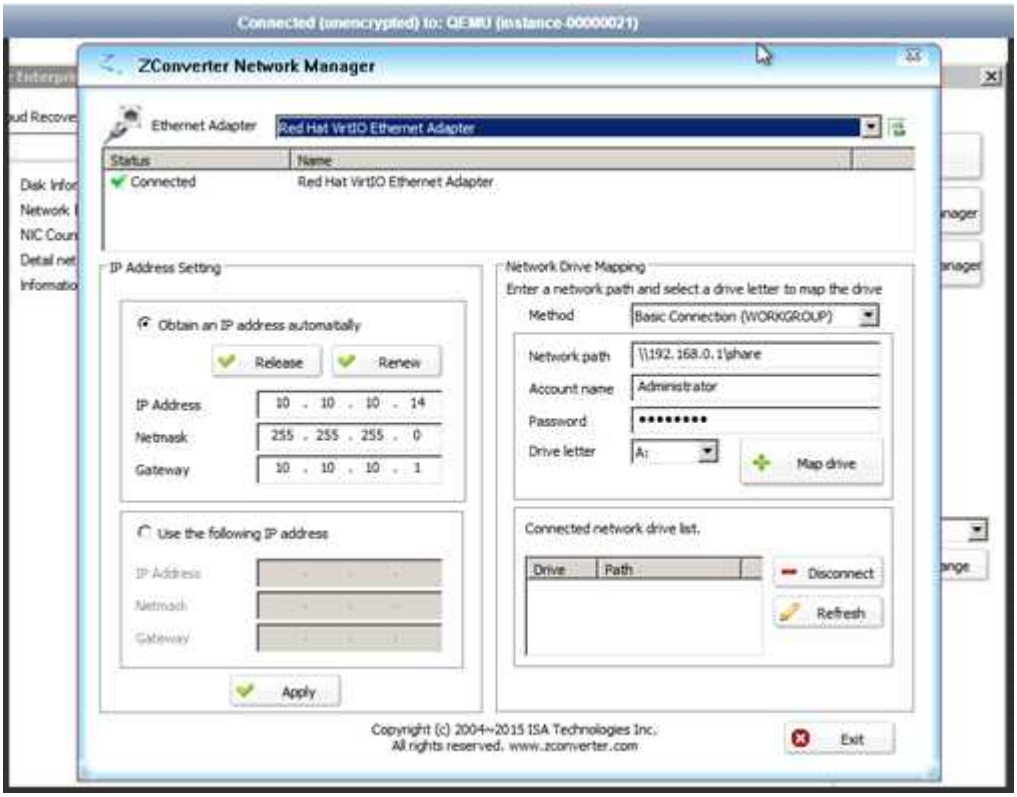
<그림 137> 에뮬레이션 변환 대상 가상 서버 준비



<그림 138> 가상 서버에 zConverter 설치

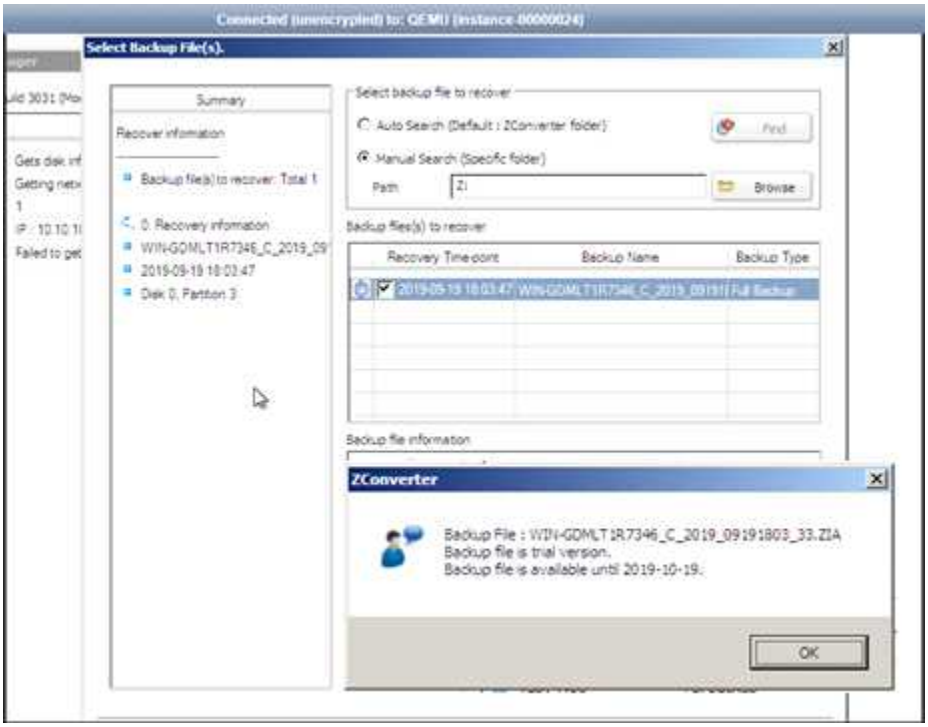
<그림 139> zConverter를 사용하여 시험 노드 1의 백업 데이터를 가상 서버로 전송

<그림 140> 가상 서버에서 zConverter로 부팅

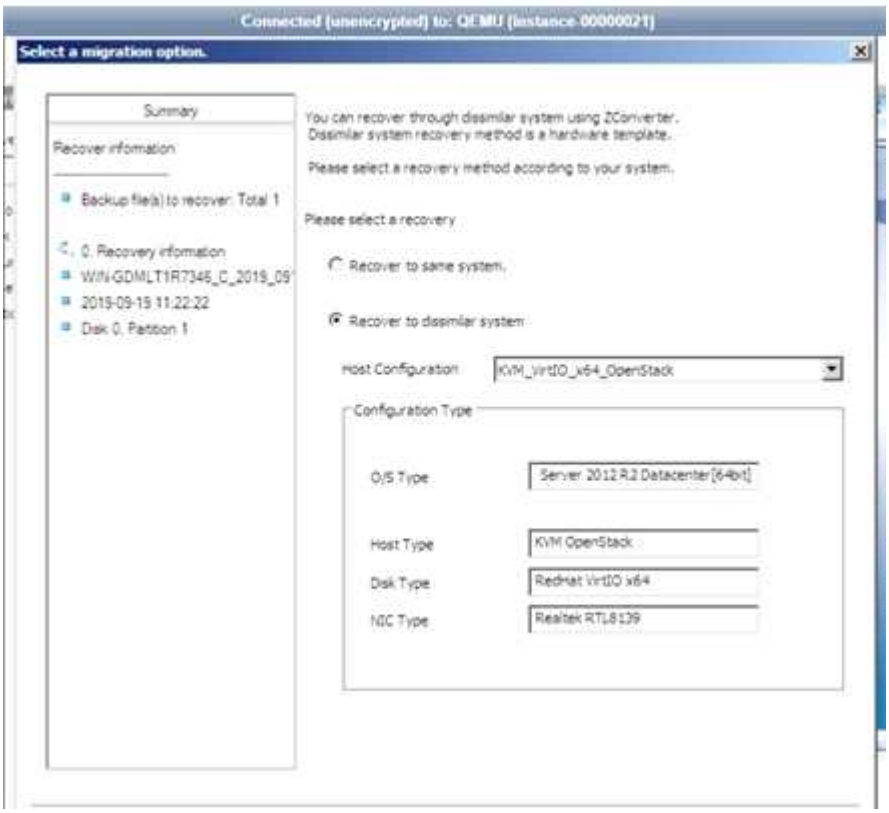


<그림 141> zConverter 가상화 작업 1: 네트워크 설정

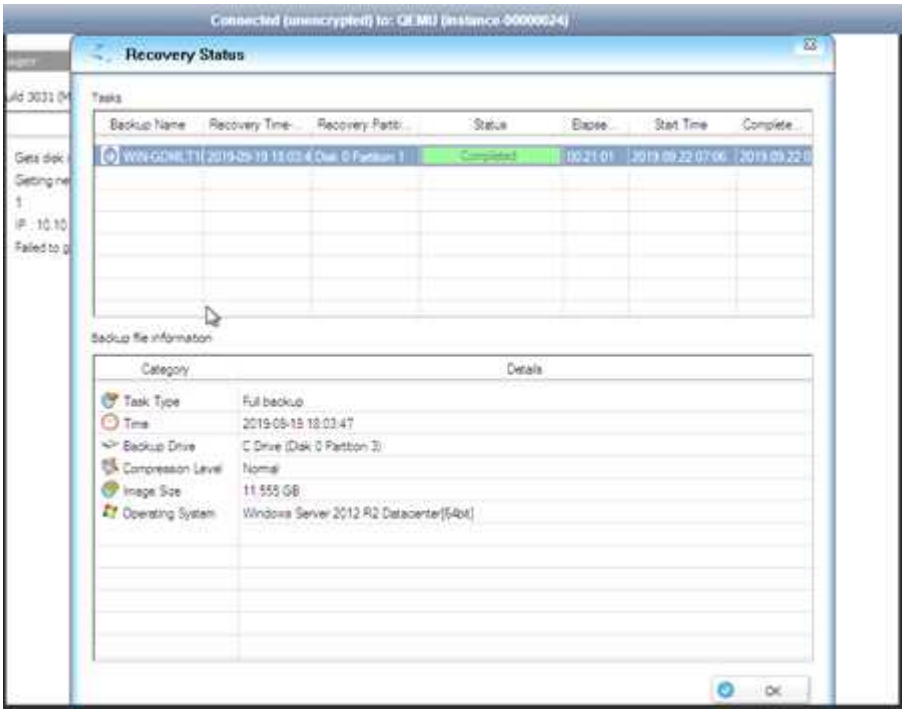
<그림 142> zConverter 가상화 작업 2: Migration Manager 실행



<그림 143> zConverter 가상화 작업 3: 백업 데이터로부터 복원

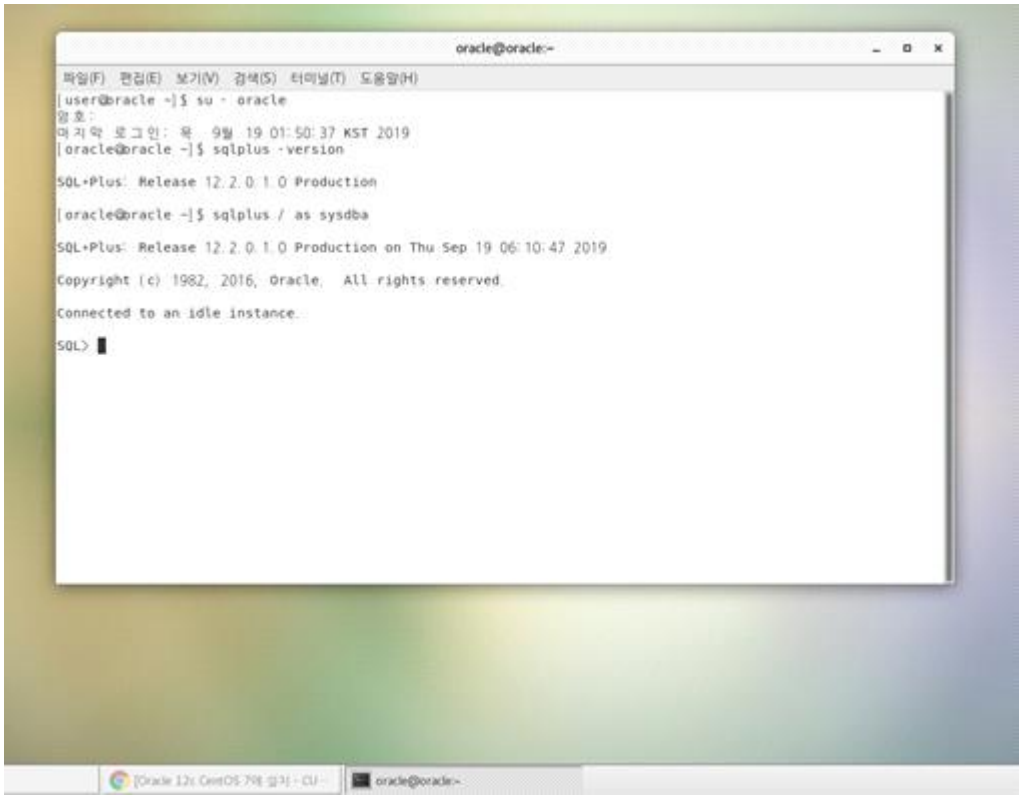


<그림 144> zConverter 가상화 작업 4: 에뮬레이션 대상 가상 서버 환경 입력

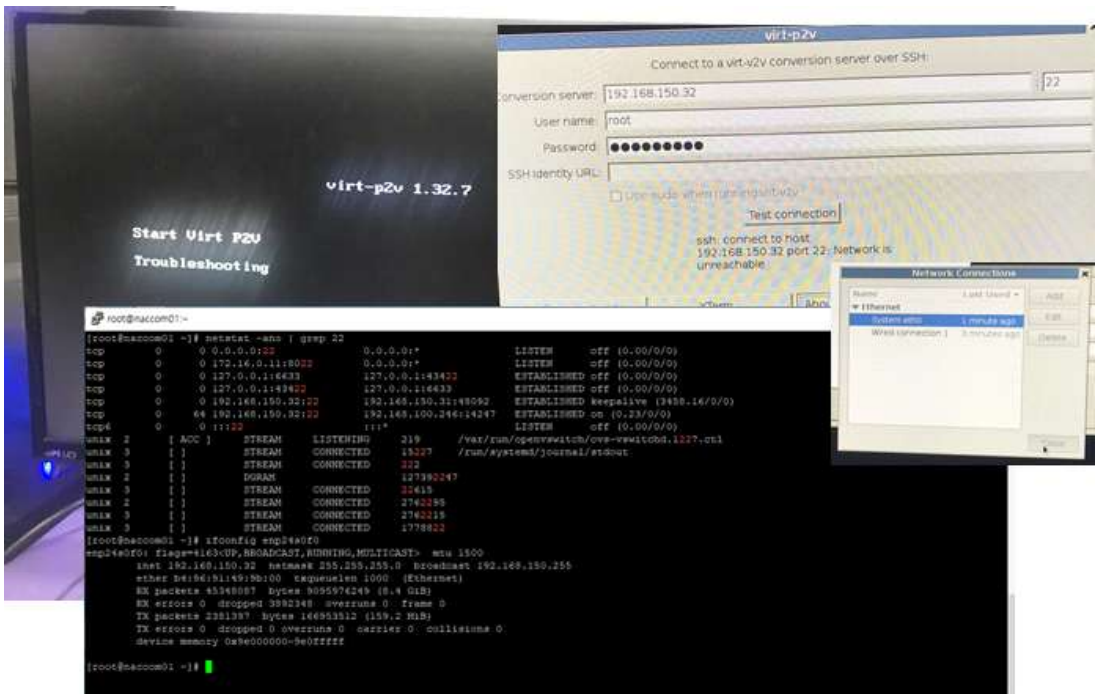


<그림 145> zConverter 가상화 작업 5: 복원 완료

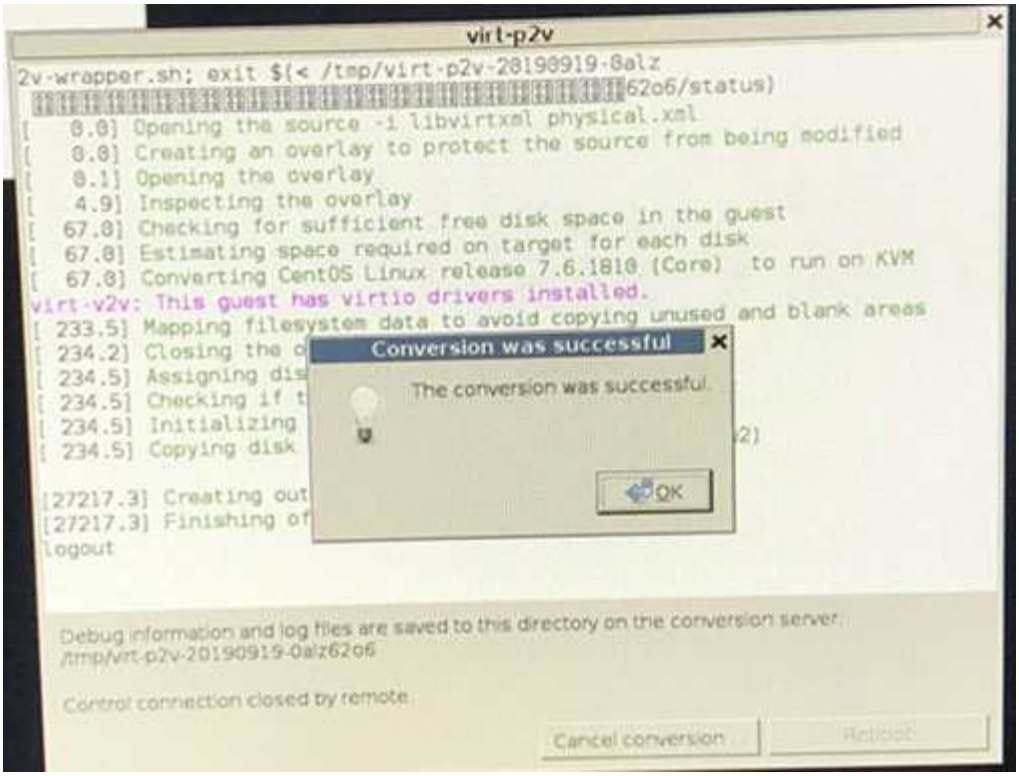
○ 시험 노드 2 (기록관리교육 DB서버)의 에뮬레이션 시험 및 검증 결과는 <그림 146 ~ 150>와 같음



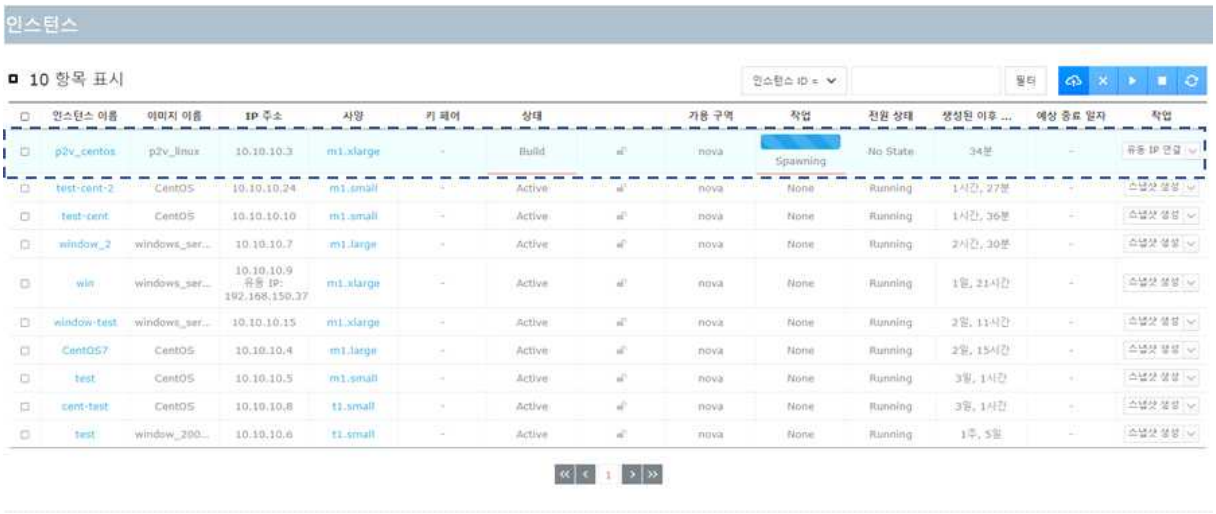
<그림 146> 시험 노드 2 환경 구성 (Oracle)



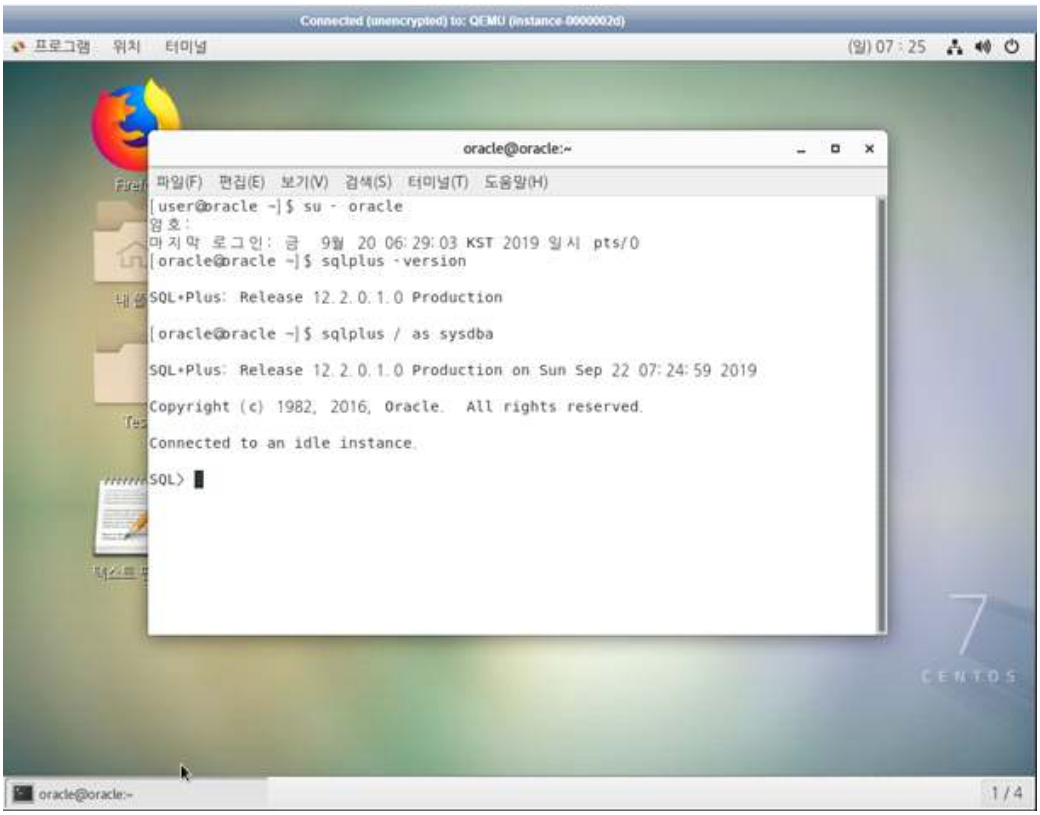
<그림 147> 시험 노드 2 중계 서버 구성



<그림 148> 시험 노드 2 virt-p2v 변환



<그림 149> 변환된 시험 노드 2의 가상 서버 생성



<그림 150> 시험 노드 2 가상 서버 콘솔 화면

3.4.2 2차 시험 및 검증: 가상 국세청 홈택스 시스템

○ 1차 시험 및 검증 이후, 실제 운영 중인 시스템을 제공 상의 문제점을 고려하여, 가상의 시스템을 개발하여 진행하기로 함

- 가상 시스템은 국세청 홈택스 웹 서비스를 선택하여, 실제 기능을 제공하는 웹 서비스가 아닌 유사하게 구현된 가상의 서비스를 개발하여 에뮬레이션 시험을 진행함

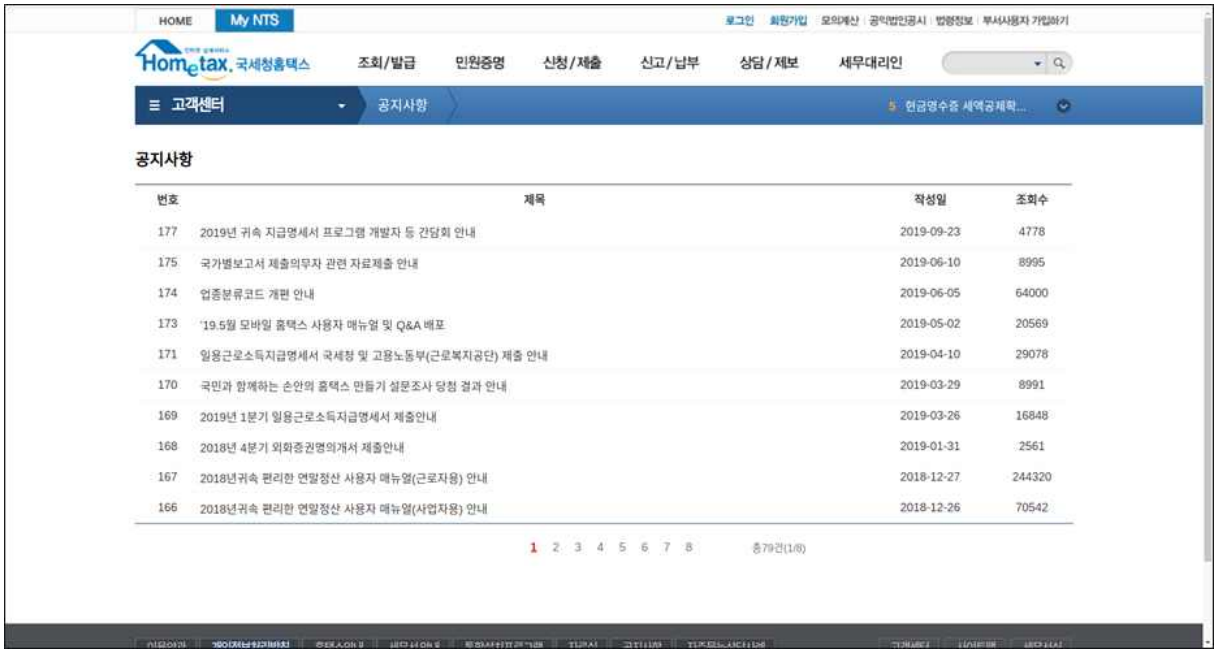
<그림 151> 국세청 홈택스 웹 서비스 화면 (www.hometax.go.kr)

○ 2차 시험 및 검증에서 사용될 가상 국세청 홈택스 시스템은 전자정부 프레임워크인 자바 스프링 프레임워크를 기반으로 개발하였으며 <그림: 가상 국세청 홈택스 서비스 개발 서버 구성>과 같이 서버를 구성함

<그림 152> 가상 국세청 홈택스 서비스 개발 서버 구성

○ 가상 국세청 홈택스 시스템 구현 서비스 화면은 아래 <그림 153 ~ 155>와 같이 세 가지 화면으로 구성하여 개발함

<그림 153> 가상 국세청 홈택스 웹 서비스 화면 1



<그림 154> 가상 국세청 홈택스 웹 서비스 화면 2

<그림 155> 가상 국세청 홈택스 웹 서비스 화면 3

○ 위에서 설명한 개발 내용을 바탕으로 2차 시험 및 검증 시스템을 <표 127>, <표 128>와 같이 구성하여 진행함

시스템	· 가상 국세청 홈택스 시스템
서버	· 가상국세청홈택스 WEB서버
운영체제	· Windows Server 2012 Standard R2 (x64)
구성요소	· Apache 2.4.41 · Tomcat 8.5.47

<표 127> 2차 시험 및 검증 시스템 시험 노드 1 구성 정보

시스템	· 가상 국세청 홈택스 시스템
서버	· 가상국세청홈택스 DB서버
운영체제	· CentOS 7.7
구성요소	· MariaDB 10.1.41

<표 128> 2차 시험 및 검증 시스템 시험 노드 2 구성 정보

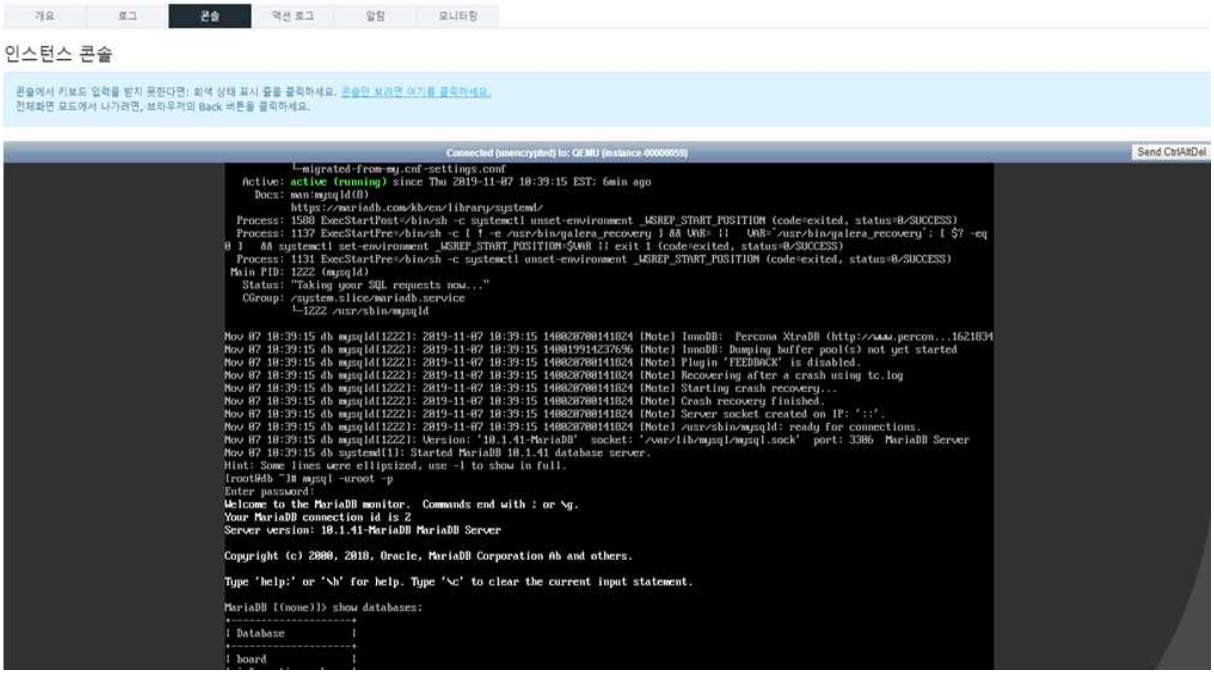
○ 각 시험 노드 1, 2를 에뮬레이션 시험 환경을 위한 포맷으로 변형 및 업로드 결과는 아래 <그림 156>와 같음

<그림 156> 2차 시험 및 검증 에뮬레이션 환경 이미지 업로드 결과

○ 에뮬레이션 시험 환경에 업로드된 이미지로 재현된 결과는 아래 <그림 157 ~ 159>과 같음

<그림 157> 2차 시험 및 검증 에뮬레이션 환경 가상 서버 생성 결과

<그림 158> 2차 시험 및 검증 시험 노드 1 가상 서버 콘솔 화면



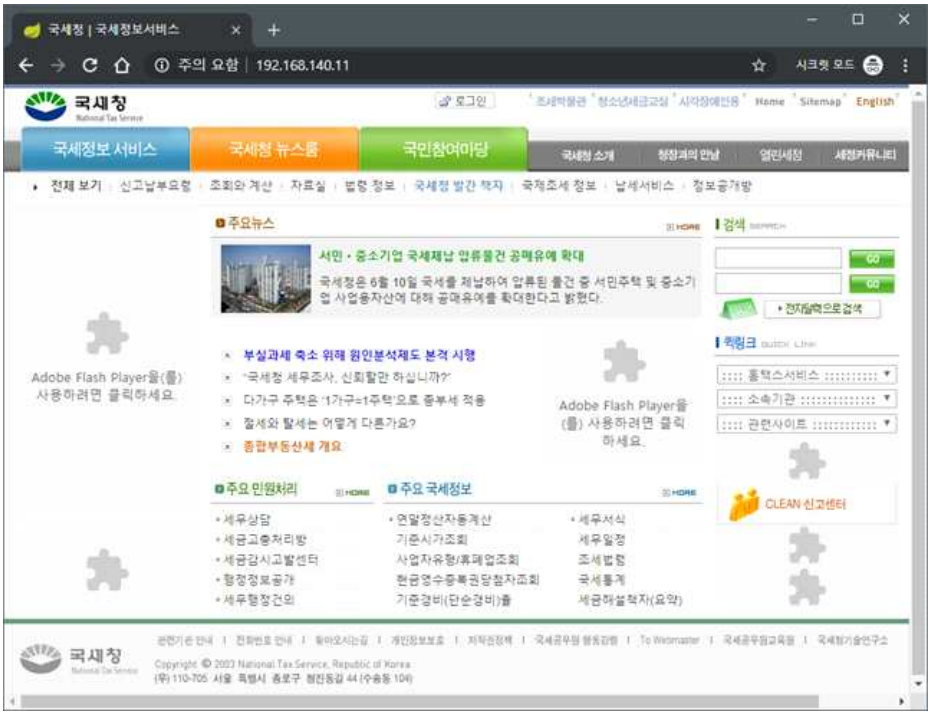
<그림 159> 2차 시험 및 검증 시험 노드 2 가상 서버 콘솔 화면

○ 2차 시험에는 시스템 구성요소 설정항목을 아래 <표 129> 와 같이 변경하였음

항목	원본시스템	대상시스템
· IP 주소	· 192.168.100.152	· 10.10.10.12
· 호스트 파일 내 IP 정보	· localhost 192.168.100.152	· localhost 10.10.10.12
· WAS 의 DB 서버 설정	· 192.168.100.106	· 10.10.10.6
· 소스수정	· 변동사항 없음	· 변동사항 없음

<표 129> 2차 시험 시스템 구성요소 설정 항목 예시

○ 추가 요청 사항으로 현재 국세청 홈택스가 아닌 과거 시스템 화면을 기반으로 하여 아래 <그림 160>와 같이 개발을 진행함

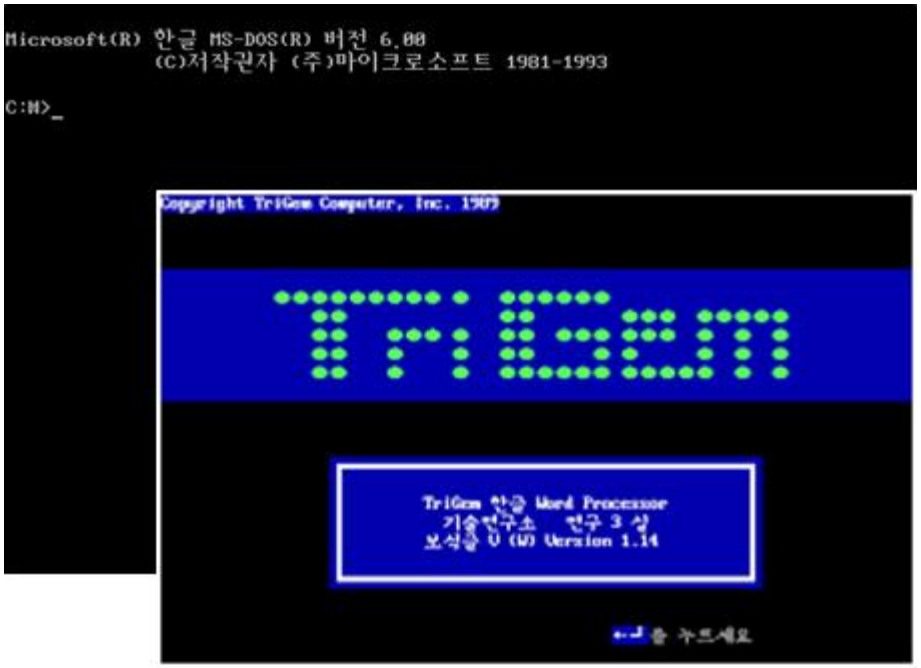


<그림 160> 가상의 2003년도 국세청 홈택스 웹 서비스 구현 화면

3.4.3 추가 시험 및 검증: MS-DOS 보석글 환경

○ 본 시험 및 검증 단계에서는 과거의 MS-DOS 환경에서 보석글 프로그램의 에뮬레이션 시험 및 검증을 진행함

- 본 단계에서는 p2v는 불가능하여, v2v로 에뮬레이션 시험 및 검증을 진행하였음



<그림 161> MS-DOS와 보석글

○ MS-DOS 라이선스는 특허 소프트웨어(Proprietary Software)이며, 현재 1.25, 2.0버전이 MIT 라이선스로 오픈소스화되어 공개됨

- 보석글 프로그램 구동을 위해 MS-DOS 6.2 한글 버전을 사용하였으나, 해당 버전에 대한 라이선스 정보는 현재 파악이 불가능함

○ 보석글은 T/Maker Reasearch라는 회사의 CP/M이라는 영문 워드 프로세서를 라이선스를 구매하여 한글화한 소프트웨어로 보석글 자체에 대한 라이선스 현황도 현재는 파악이 불가능함

○ 에뮬레이션 시험 및 검증을 위해 필요한 원본 서버는 Virtual Box를 이용하여 생성하며, 상세한 생성 방법은 다음과 같음

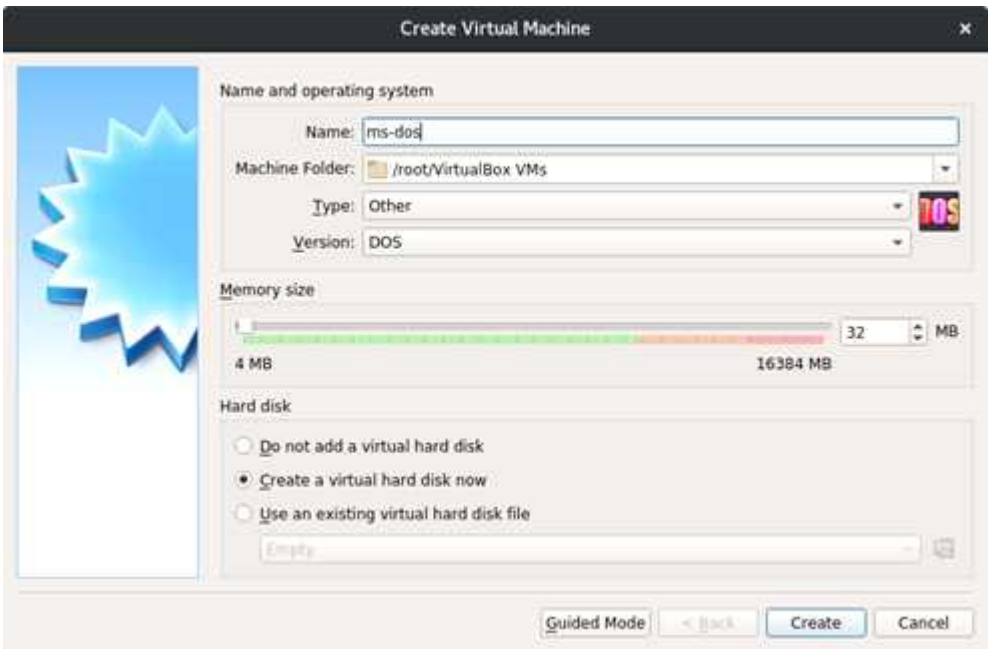
- 먼저 가상 서버 이미지를 생성하기 위해 VirtualBox를 실행

<그림 162> MS-DOS & 보석글 이미지 제작 방법: VirtualBox Manager 실행

- VirtualBox Manager에서 가상 서버 실행 버튼을 클릭하여, 가상 서버 생성 위자드를 실행하고, 전문가 모드(Expert Mode)를 선택

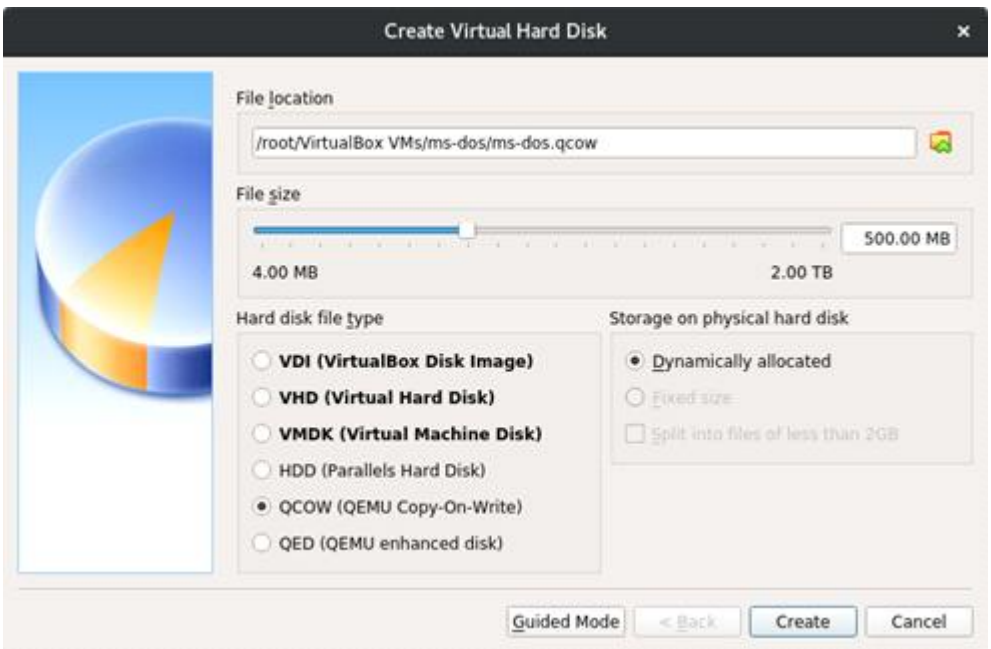
<그림 163> MS-DOS & 보석글 이미지 제작 방법: 가상 서버 생성 위자드 실행

- 가상 서버의 이름, 저장 경로, 유형, 버전 등의 기본 정보를 아래와 같이 입력



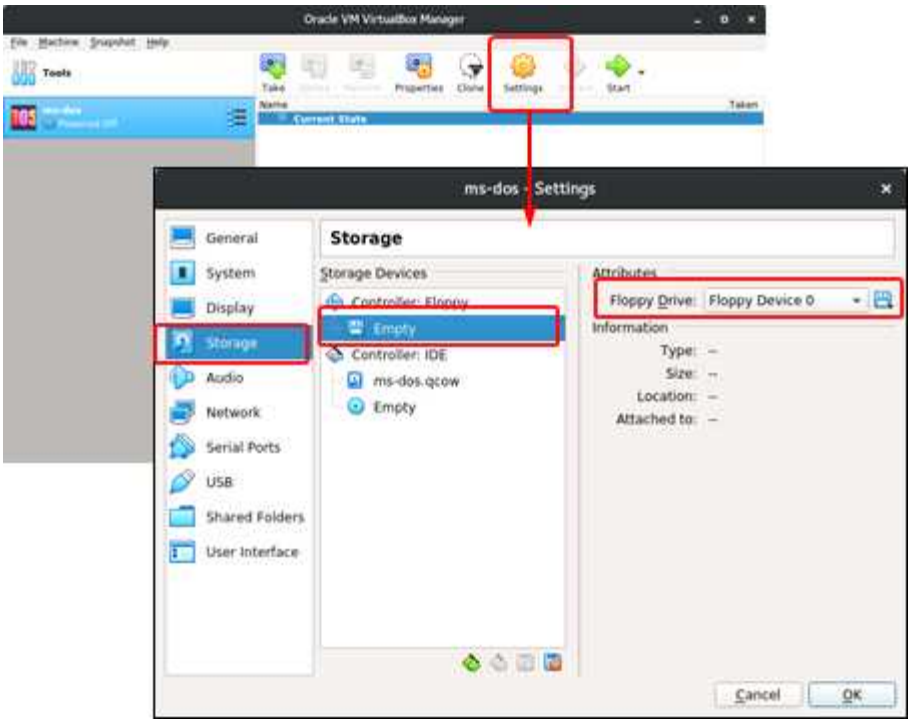
<그림 164> MS-DOS & 보석글 이미지 제작 방법: 가상 서버 기본 정보 입력

- 가상 서버의 이미지 파일에 대한 상세 정보를 입력하여 가상 서버 이미지를 생성



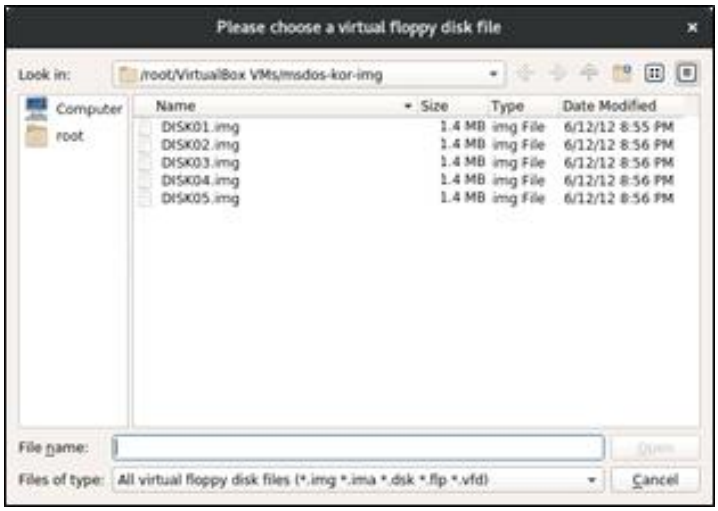
<그림 165> MS-DOS & 보석글 이미지 제작 방법: 가상 서버 상세 정보 입력

- 생성된 가상 서버에 MS-DOS 및 보석글 설치 이미지를 추가하는 방법은 아래와 같으며, 먼저 MS-DOS 이미지를 추가하여 설치하고, 보석글 이미지를 추가하는 순서로 진행



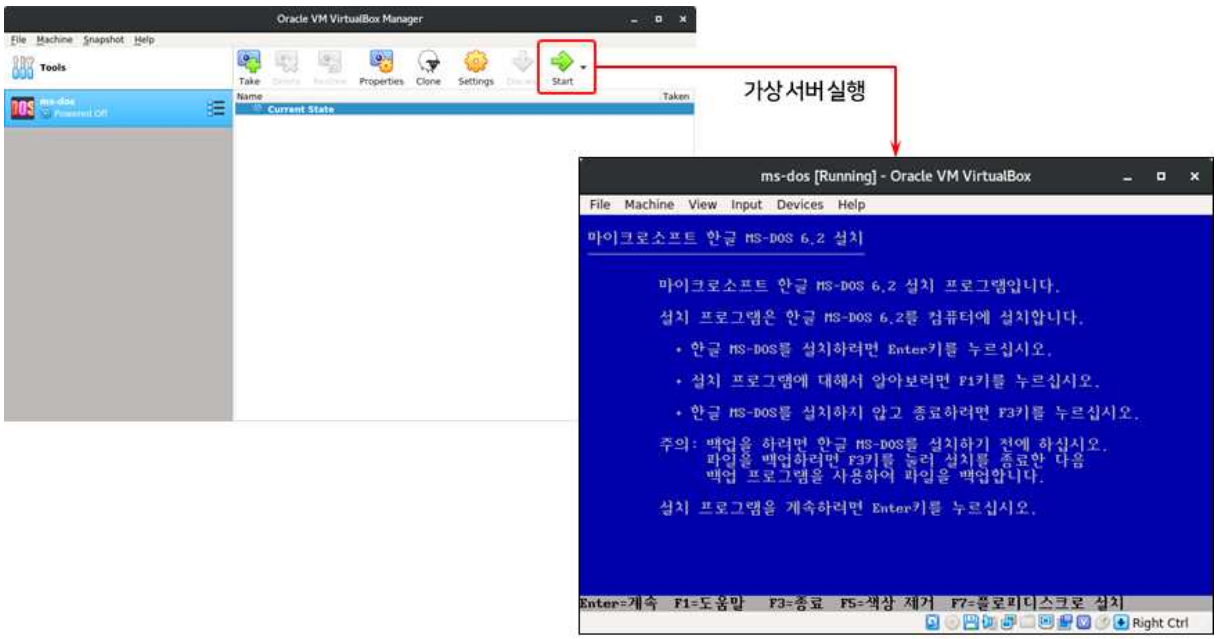
<그림 166> MS-DOS & 보석글 이미지 제작 방법: 가상 서버 플로피 디스크 추가 방법

- 인터넷에서 다운로드 받은 MS-DOS 한글 6.2 버전 이미지의 디스크 1을 추가



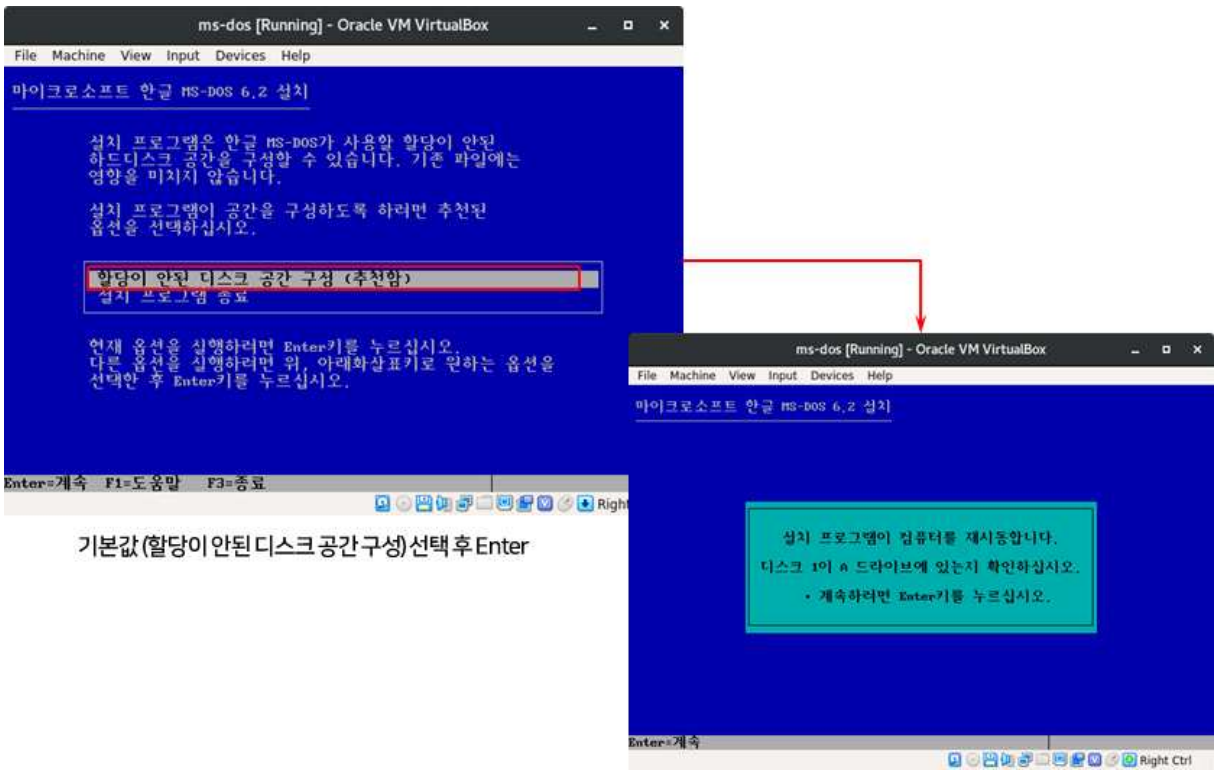
<그림 167> MS-DOS & 보석글 이미지 제작 방법: MS-DOS 디스크 추가

- 디스크 추가 후, 생성된 가상 서버를 다음과 같이 실행하고, MS-DOS를 설치

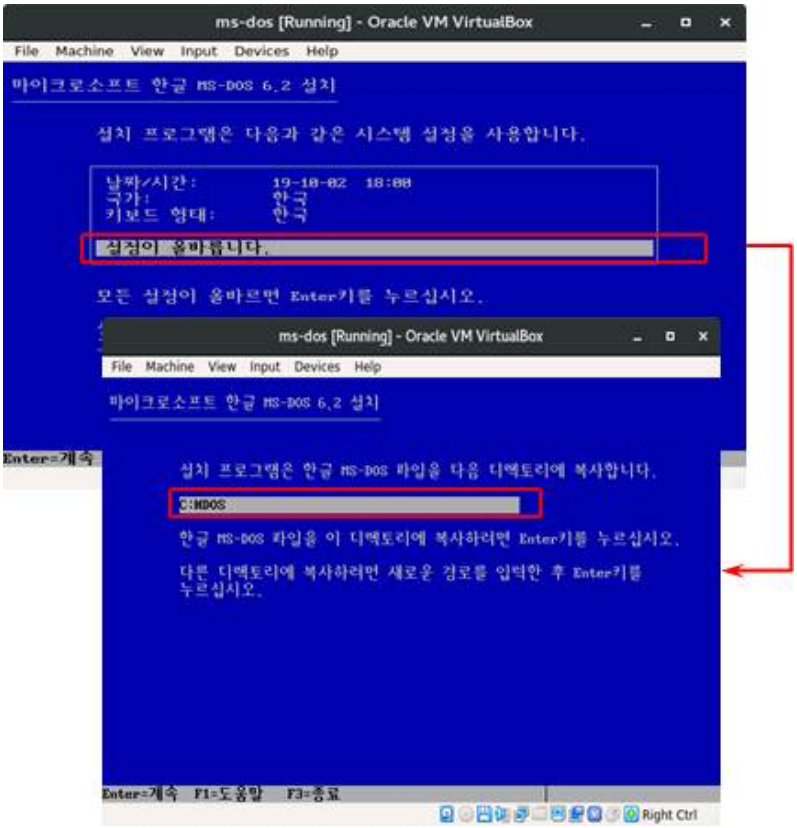


가상서버실행화면

<그림 168> MS-DOS & 보석글 이미지 제작 방법: 가상 서버 실행

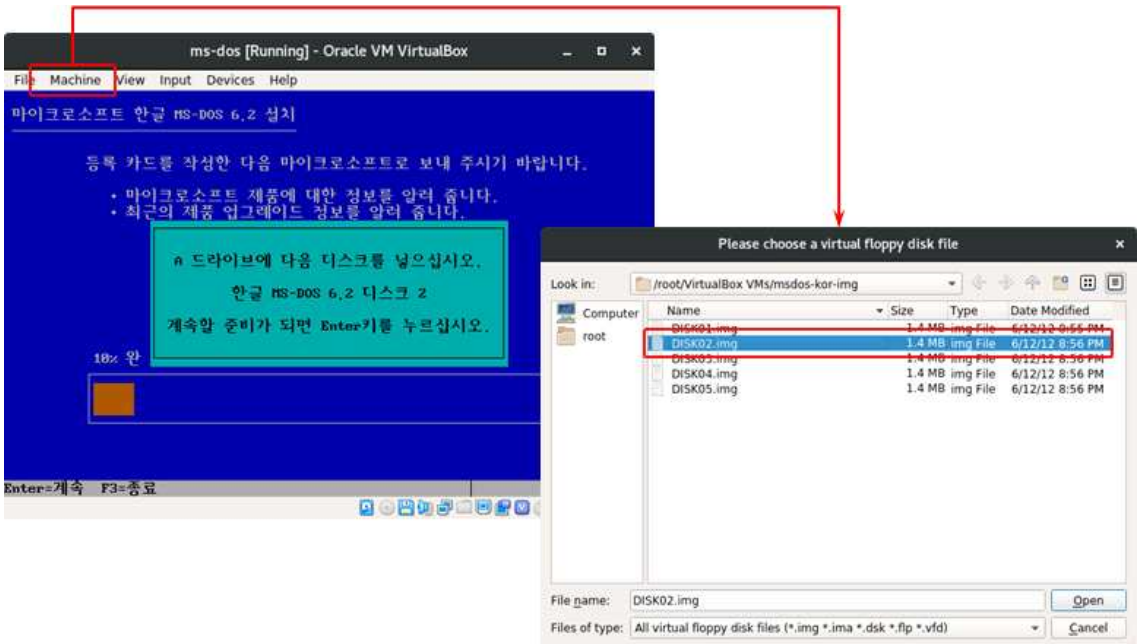


<그림 169> MS-DOS & 보석글 이미지 제작 방법: MS-DOS 설치 1

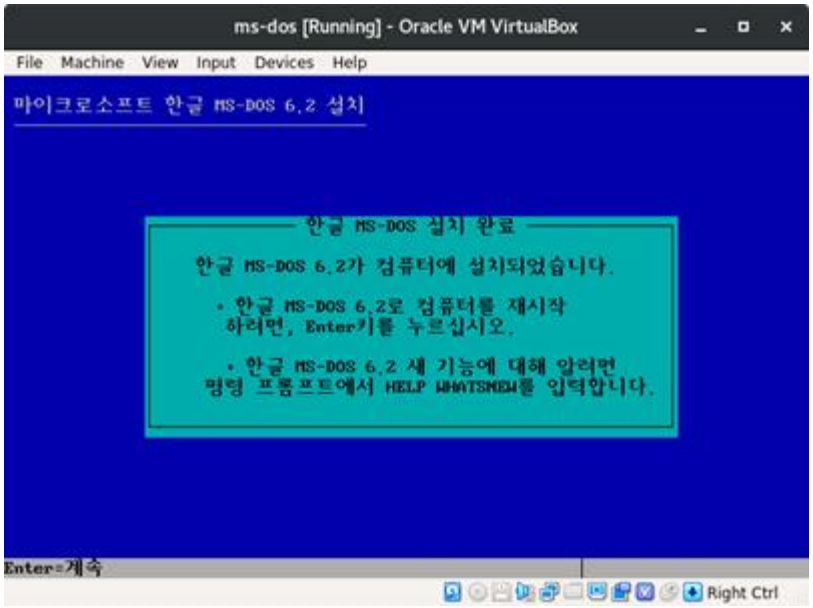


<그림 170> MS-DOS & 보석글 이미지 제작 방법: MS-DOS 설치 2

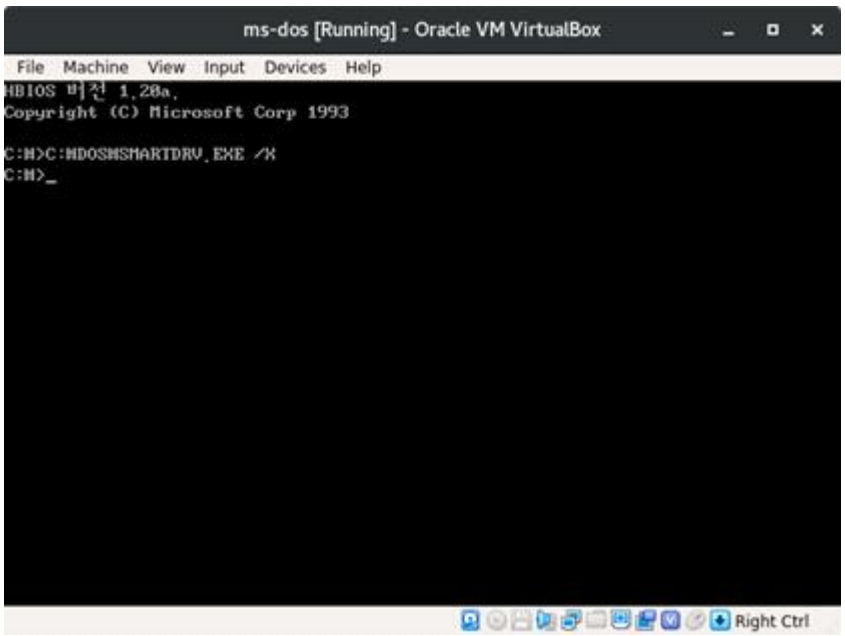
- MS-DOS 설치 중 디스크 추가 메시지가 뜨면 다음과 같이 디스크를 추가하여 진행 (디스크 3, 4, 5도 동일한 방법으로 진행)



<그림 171> MS-DOS & 보석글 이미지 제작 방법: MS-DOS 설치 3

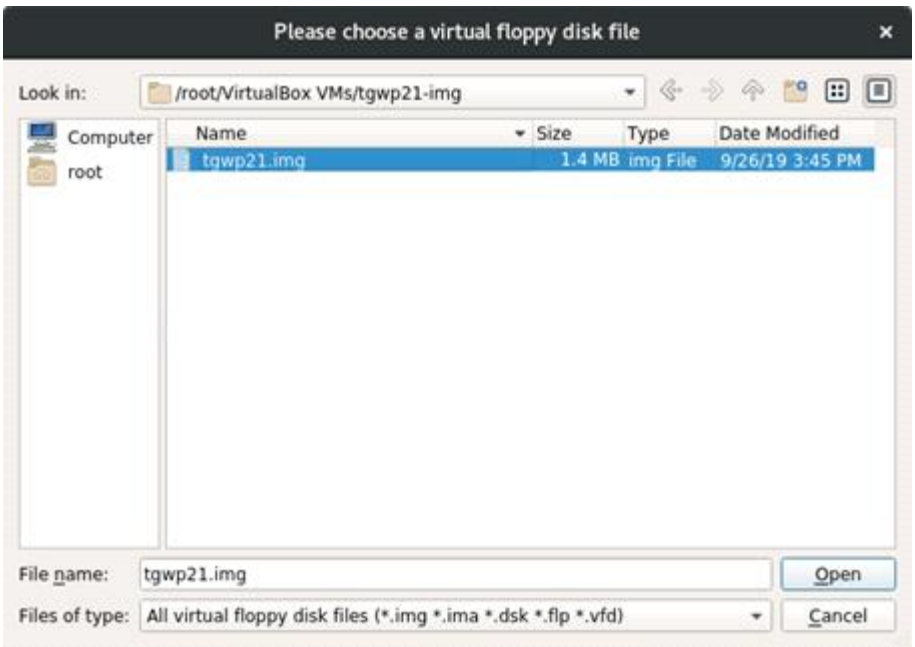


<그림 172> MS-DOS & 보석글 이미지 제작 방법: MS-DOS 설치 완료



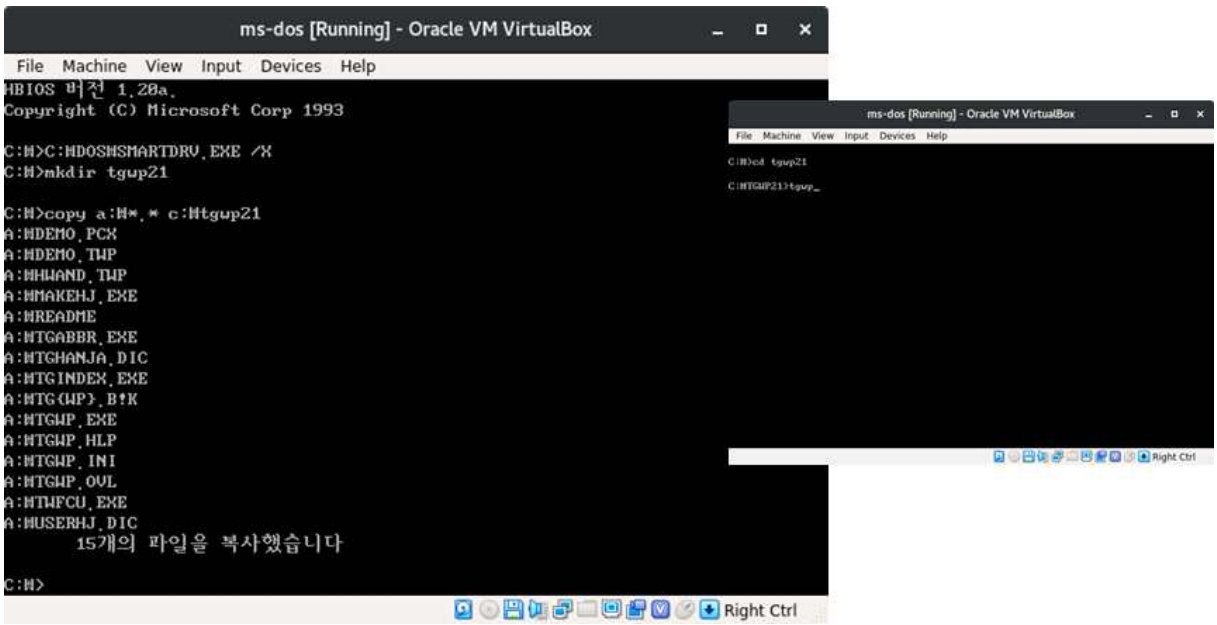
<그림 173> MS-DOS & 보석글 이미지 제작 방법: MS-DOS 실행

- MS-DOS 설치 완료 후, 보석글 이미지 디스크를 추가



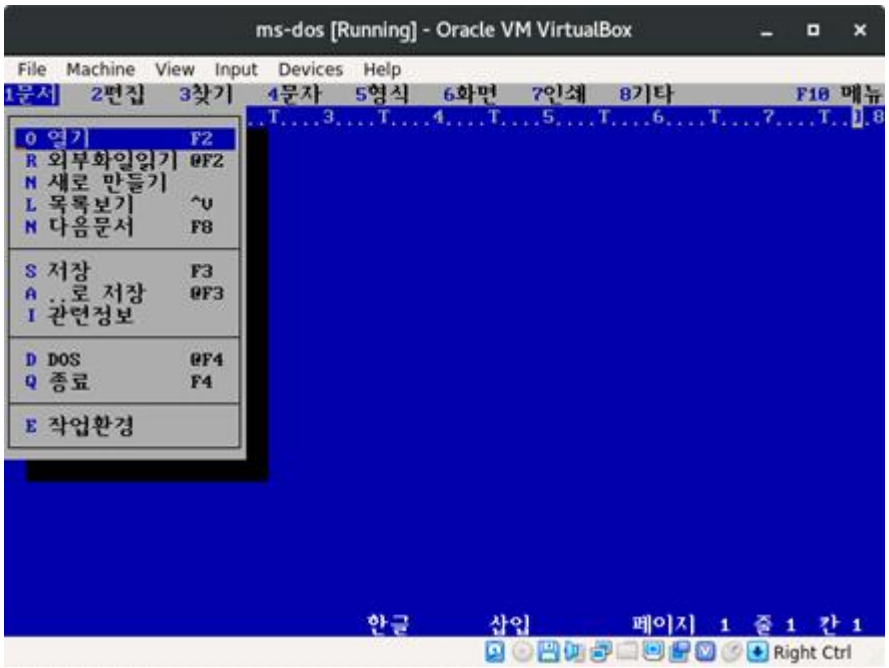
<그림 174> MS-DOS & 보석글 이미지 제작 방법: 보석글 이미지 추가

- 보석글 이미지 내 파일을 MS-DOS의 파일 시스템으로 복사



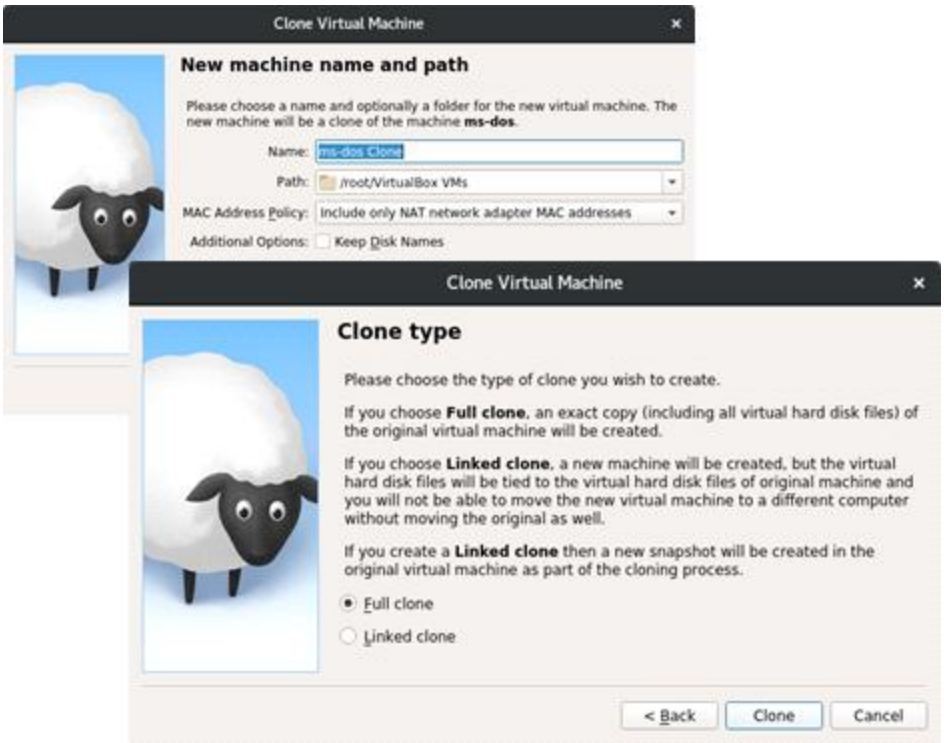
<그림 175> MS-DOS & 보석글 이미지 제작 방법: 보석글 이미지 파일 복사

- 복사한 경로에서 TGWP.EXE 파일 실행 (실행 파일 이름은 원본 보석글 이미지 디스크 내의 내용에 따라 다를 수 있음)



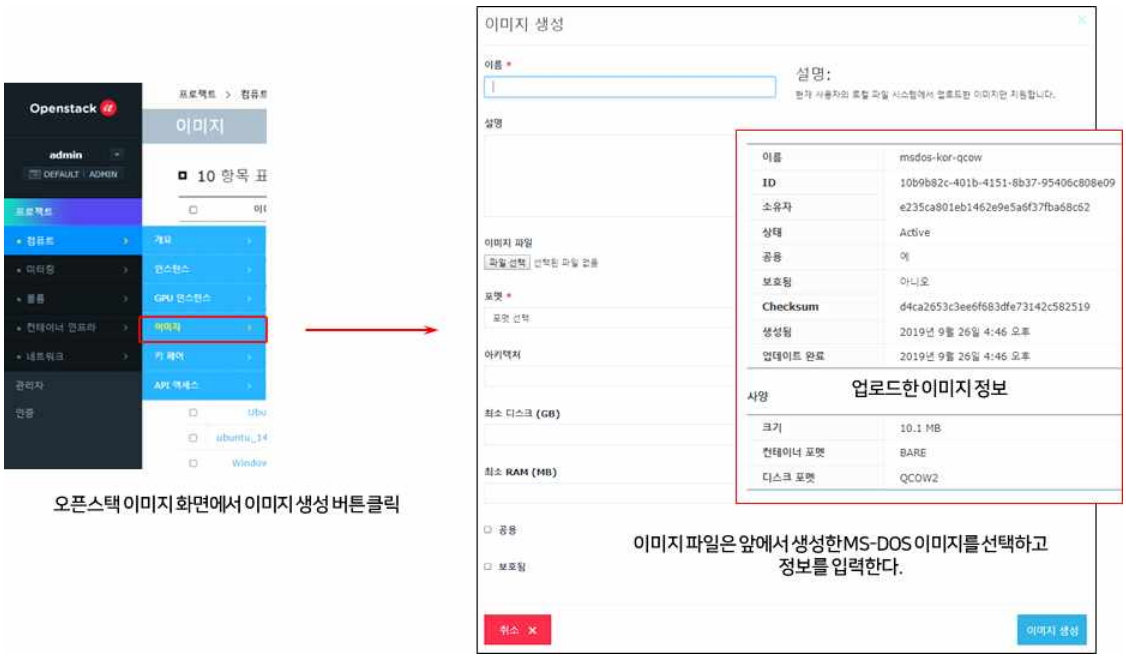
<그림 176> MS-DOS & 보석글 이미지 제작 방법: 보석글 실행 화면

- 설치 완료를 확인하고, 가상 서버를 종료한 뒤에 가상 서버 이미지를 복제 (복제된 이미지가 아니면 에뮬레이션 시험 환경에서 구동되지 않음)



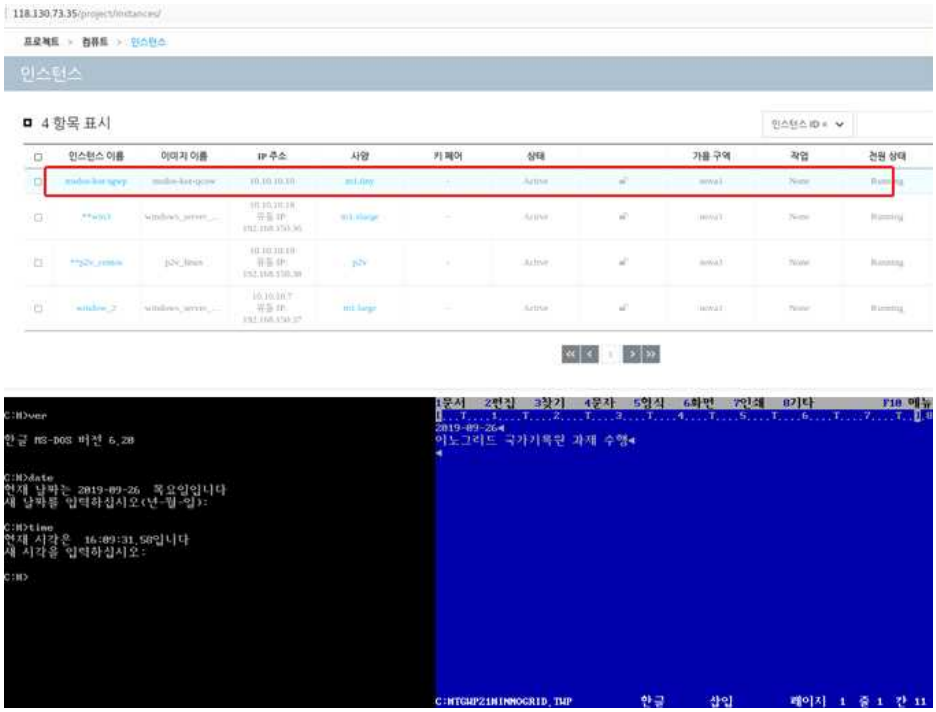
<그림 177> 생성된 가상 서버 이미지 복제

- 생성된 가상 서버 이미지 복제본을 에뮬레이션 시험을 위한 환경에 업로드



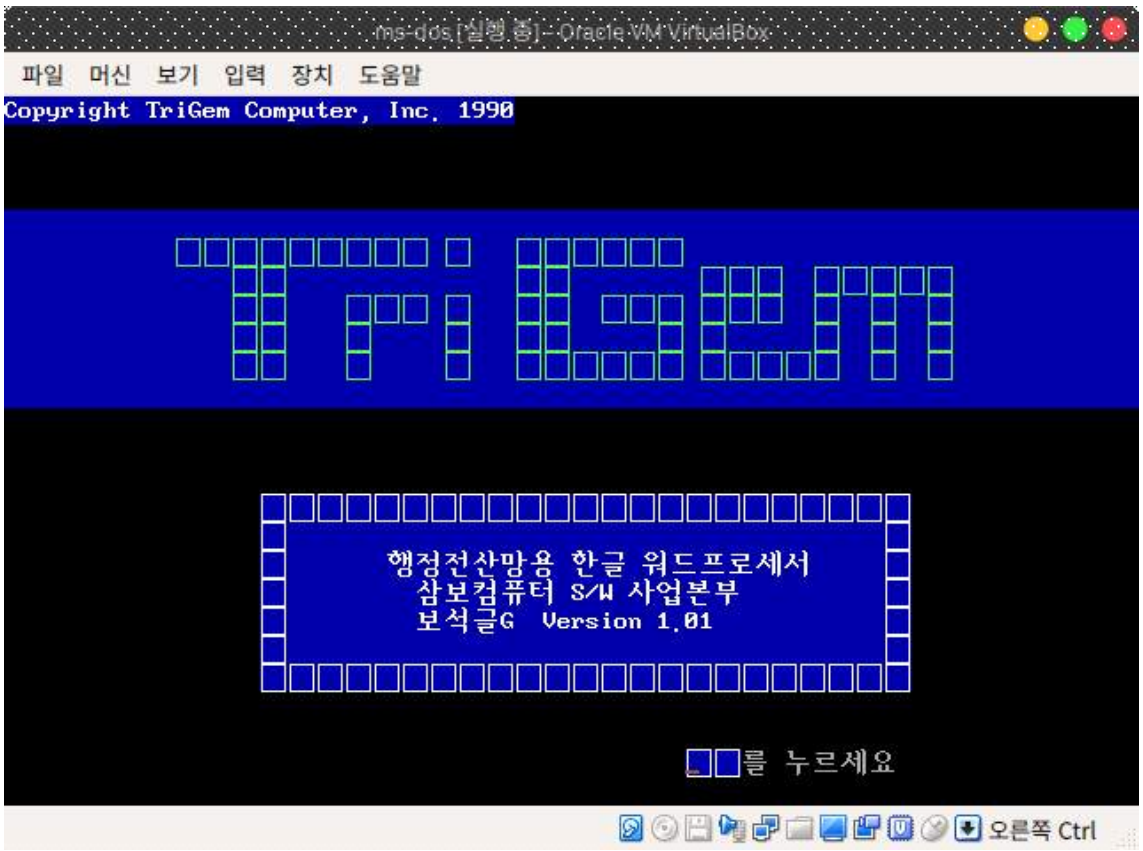
<그림 178> 에뮬레이션 시험 환경에 대상 서버 이미지 업로드

○ 위에서 생성한 이미지로 진행한 에뮬레이션 시험 및 검증 결과는 다음과 같음



<그림 179> MS-DOS&보석글 에뮬레이션 결과 화면

○ 추후 요청사항으로 행정전산망용 보석글 버전에 대해서도 동일하게 진행함



<그림 180> 행정전산망용 보석글 에뮬레이션 화면

3.5 에뮬레이션 절차 및 정합성 검증 항목 도출

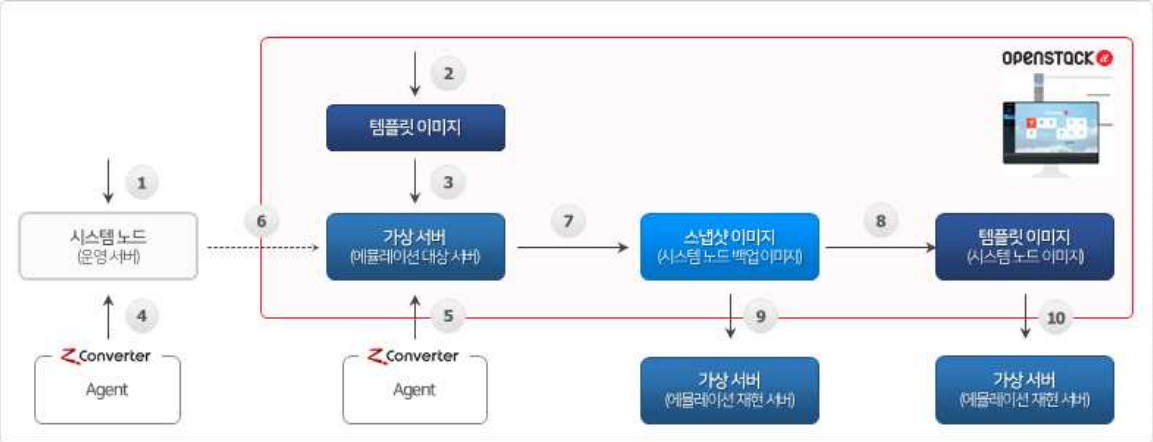
- 에뮬레이션 대상 시스템 선정 및 에뮬레이션 시험 과정을 통해 에물레이션을 진행하기 위한 절차를 정의함
- 에뮬레이션 시험이 원래 그대로 외관(Look & Feel)으로 재현하는 것을 검증하기 위해서 정합성 항목을 도출함

3.5.1 에뮬레이션 절차 정의

○ 선정된 대상 시스템에 대한 에뮬레이션 시험을 진행하기 위한 에뮬레이션 절차를 정의함

- 에뮬레이션 시험을 진행하기 위해서는 어떻게 진행할 것인지에 대한 절차가 필요하며, 이에 대한 에뮬레이션 절차 정의가 필요함
- 에뮬레이션 시험은 p2v, v2v 두 가지 방식으로 진행함

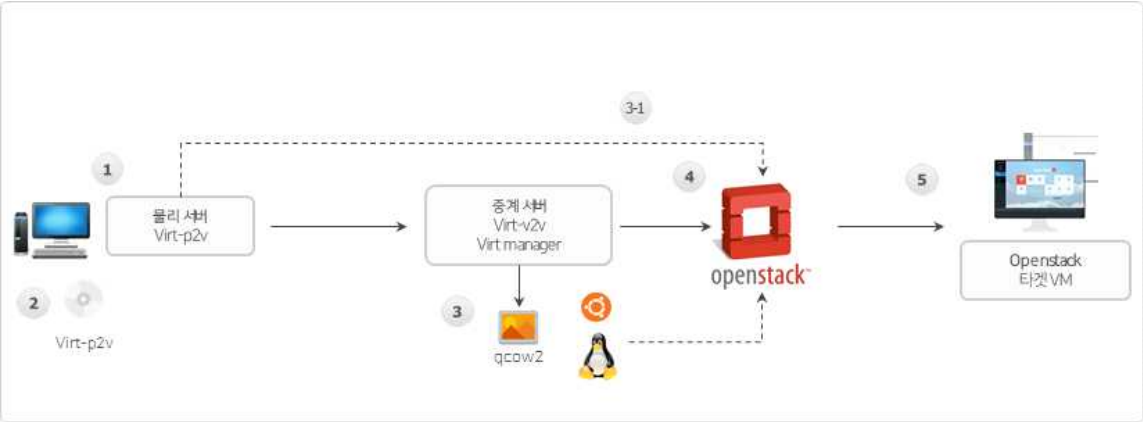
○ 본 연구에서 진행한 p2v 방식의 에뮬레이션은 zConverter, virt-p2v 두 가지 도구를 사용하여 진행하며, 아래에 제시된 <그림 181>은 각 도구를 사용하였을 때의 절차를 나타나며, <표 130>은 단계별 상세 내용을 설명함



<그림 181> zConverter 기반 에뮬레이션 시험 절차

단계	설명
1	· 에뮬레이션을 하고자 하는 대상 시스템의 노드(운영서버)를 준비한다.
2	· 운영 서버와 같은 운영체제로 제작된 템플릿 이미지를 준비한다.
3	· 제작된 템플릿 이미지로 가상서버(에뮬레이션 대상 서버)를 생성한다.
4	· 운영 서버에 zConverter Agent를 설치한다.
5	· 에뮬레이션 대상 서버에도 zConverter Agent를 설치한다.
6	· zConverter 도구를 사용하여 운영 서버에서 에뮬레이션 대상 서버로 데이터를 전송한다. (두 서버가 서로 통신이 가능해야 한다.)
7	· 운영서버가 에뮬레이션 대상 서버로 정상적으로 변환되었는지 확인하고, 스냅샷 기능을 통해 백업본을 생성한다.
8	· 백업된 스냅샷 이미지를 템플릿 이미지(시스템 노드 이미지)로 변환한다.
9	· 백업된 스냅샷 이미지로부터 가상 서버를 생성한다.
10	· 생성된 시스템 노드 이미지로부터 가상 서버를 생성한다.

<표 130> zConverter 기반 에뮬레이션 절차 단계



<그림 182> virt-p2v 기반 에뮬레이션 시험 절차

단계	설명
1	· 에뮬레이션을 하고자 하는 대상 시스템의 노드(운영서버)를 준비한다.
2	· 운영 서버에 virt-p2v로 부팅하여 중계 서버에 이미지를 컨버팅 하여 전송한다.
3	· 중계 서버에 virt-p2v, virt manager를 설치하여 이미지를 확인 후 다운로드 한다.
3-1	· 3번 단계를 생략하고, 중계 서버 없이 직접 virt-p2v, virt manager를 설치도 가능하다.
4	· 오픈스택에 이미지를 업로드한다.
5	· 업로드된 이미지로 가상 서버를 생성한다.

<표 131> virt-p2v 기반 에뮬레이션 절차 단계

○ 본 연구에서 v2v 방식의 에뮬레이션은 kvm 기반의 qcow2 가상 이미지 기반으로 VirtualBox와 VirtManager 도구를 활용하며, 상세한 절차는 <그림 183> 및 <표 132>에서 설명함

<그림 183> v2v 에뮬레이션 시험 절차

단계	설명
1	· 가상 서버 생성 도구를 활용하여 kvm 기반의 qcow2 가상 서버 이미지를 생성한다.
2	· 생성된 이미지를 에뮬레이션 시험 환경인 오픈스택에 업로드한다.
3	· 업로드된 이미지로 가상 서버를 생성한다.

<표 132> v2v 기반 에뮬레이션 절차 단계

3.5.2 에뮬레이션 정합성 검증 항목 도출

○ 본 연구의 에뮬레이션은 원본 시스템을 원래 그대로의 외관(Look & Feel)으로 재현하는데 그 목적이 있음

- 에뮬레이션 과정 중에 발생하는 시스템 상의 변형(하드웨어의 종류 및 사양, 네트워크 정보 등)이 발생하더라도 외관상 기존과 동일하게 동작 시키는 것을 목표로 함

- 총 6개의 운영체제와 8개의 소프트웨어에 대한 에뮬레이션 정합성을 검증함

검증 항목		기존 시스템	신규 시스템
운영체제	MS-DOS 6.2 한글	정상	정상
	Windows XP	정상	정상
	Windows Server 2012 Standard R2 (x64)	정상	정상
	CentOS 7.6 1810	정상	정상
	CentOS 7.7 1908	정상	정상
	Ubuntu 14.04	정상	정상
소프트웨어	보석글V(W) 1.14	정상	정상
	보석글G 1.01 (행정전산망용)	정상	정상
	한글 97	정상	정상
	MySQL 5.5	정상	정상
	Oracle 12g	정상	정상
	WebtoB v4.1	정상	정상
	JEUS v6.0	정상	정상
	가상 국세청 홈택스 자바 스프링 웹 앱 (Apache 2.4.41 & Tomcat 8.5.47)	정상	정상

<표 133> 에뮬레이션 정합성 검증을 위한 테스트 항목 도출

제4장 제2세부연구개발과제의 연구결과 고찰 및 결론

1. 기대효과

○ (기술적) 클라우드 기반 전자기록 보관 시스템 관련 핵심 기술 확보

- 클라우드 기반의 전자기록 저장 시스템 구성을 통한 안정적인 환경 및 보존 기술 확보
- 클라우드 스토리지를 통한 데이터 이중화, 장애복구 등 데이터 보존 신뢰성 향상 기술 확보

○ (경제적) 전자기록물 보관 및 데이터 제공 서비스에 필요한 자원을 효율적으로 제공하여 시스템 운영비용 절감 등 경제성 제공

- 저장 공간 사용량에 따른 유연한 확장을 지원하여 향후 데이터 증가 시 필요한 자원만큼 확보 가능
- 필요한 데이터를 제공을 위한 서비스를 가상화된 환경을 통해 제공하므로 사용자 수에 따른 유연한 서비스 제공 가능

○ (사회적) 행정정보, 전자기록물 등 중요한 데이터의 장기적 보관을 위한 체계적인 방안 마련 및 활용 서비스로의 연동을 위한 기반 환경 조성

- 데이터세트 유형의 전자기록의 장기보존 전략 수립을 구체화하는 방안 마련에 활용
- 테스트베드 구축 및 시험을 통해 장기보존에 적합한 보존방식을 검증하고 향후 DB유형 전자기록물 이관 및 보존에 기여
- 필요한 데이터를 제공을 위한 클라우드 기반 연계 및 확장 가능성을 제공하여 새로운 서비스로의 발전 가능성 제공

2. 활용 방안

○ 다양한 유형의 전자기록(표준전자문서, 시청각기록물, 웹기록물 등)에 장기보존정책 및 전략에 활용

○ 현재 행정정보시스템에서 생산되는 데이터세트에 대한 이관・보존・활용 전략 및 기술 확보

○ 디지털 컴포넌트 유형별 렌더링 방법 마련 및 에뮬레이션 전략 적용 타당성 검증을 위한 에뮬레이터 프로토타입 개발을 통한 본 디지털(born digital) 기록물을 원본 기반의 장기보존 및 활용성 고취

○ 공공기관에서 생산・관리되는 DB 유형의 전자기록에 적합한 장기보존방식을 제시하여 보

존·활용 기반 마련

○ 데이터 저장 및 관리가 필요한 민간 기업, 연구소 등에 구축하여 체계적이고 안정적인 기록물 관리 서비스에 활용

3. 사업 확장 가능성

○ (체계적인 에뮬레이션 체계 구축) 본 사업에서는 안정적인 데이터 이전 체계를 수립 및 다양한 솔루션 사용을 고려한 전자기록물 이전 환경을 고려하여 다양한 상용 솔루션 및 오픈소스를 활용할 수 있도록 전체 프로세스 설계

- 일반적인 클라우드 및 가상화 솔루션, 오픈소스 클라우드 프로젝트를 통해 생성된 가상 환경 기반 데이터 이전 고려

- P2V 상용 솔루션 및 무료 도구 등 다양한 소프트웨어 이용한 데이터 이전 환경 고려

구분	본 사업에서 활용한 솔루션		대체 솔루션	
클라우드 & 가상화 솔루션	상용	오픈스택잇	상용	클라우드잇
				vmware
				기타 솔루션
	무료	오픈스택	무료	기타 오픈소스 SW
데이터 이전 솔루션	상용	zconverter	상용	기타 솔루션
	무료	virt-p2v	무료	기타 오픈소스 SW

<표 134> 타 솔루션 및 소프트웨어로 대체 가능한 항목

제5장 제2세부연구개발과제의 연구성과

1. 활용성과

세부과제명	클라우드 기반 전자기록의 장기보존기술개발 테스트베드 구축 및 에뮬레이션 시험•검증
세부과제책임자	조철용/ (주)이노그리드 / 컴퓨터과학

가. 연구논문

· 특기사항 없음

나. 학술발표

· 특기사항 없음

다. 지적재산권

· 특기사항 없음

라. 정책활용

· 공공기관의 데이터세트 유형 전자기록 보존 및 활용을 위한 지침 수립 지원
· 전자기록물의 에뮬레이션 방식 보존 방안 및 절차 지원

마. 타연구/차기연구에 활용

· 국가기록원 데이터세트 유형의 에뮬레이션 전략을 위한 기술규격 표준화 연구 및 개발에 활용

바. 언론홍보 및 대국민교육

· 특기사항 없음

사. 기타

· 특기사항 없음

2. 활용계획

○ 안정적인 기록물 보관 및 관리가 필요한 공공기관

- 공공기관 및 국가기관에서 보유한 중요한 전자기록물의 안정적인 보관 및 관리를 위한 데이터 장기보존 시스템으로 활용
- 오래된 전자기록물을 저장 관리하고 활용하기 위한 시스템 구축에 활용
- 오래된 운영체제 및 소프트웨어에서 생성된 데이터의 보관, 기록 확인을 위한 시스템에 활용

○ 데이터 기록 관리가 필요한 민간기업

- 민간 기업의 기술 개발 문서, 계약 문서 등 다양한 전자기록물을 보관하고 관리하기 위한 전자기록물 관리 시스템에 활용
- 4차 산업혁명 시대에 급증하는 데이터로 인해 발생한 전자기록물을 효과적으로 저장하고 관리하기 위한 저장소에 활용
- 이직, 퇴사 등으로 소멸하는 전자기록물을 효과적으로 저장하고 추후 확인하기 위한 저장 공간으로 활용

○ 다양한 정보 저장, 제공하는 데이터 비즈니스 기업

- 데이터 비즈니스 기업이 보유한 전자기록물 정보를 효과적으로 관리하기 위한 시스템으로 활용
- 다양한 형태의 정보로 가공된 데이터를 저장하고 애뮬레이션하여 테스트하기 위한 환경으로 활용
- 데이터 제공을 위한 가상 환경 구축 및 서비스 운영에 활용

○ 클라우드, 스토리지, 데이터 이전 관련 사업자

- 다양한 기관의 클라우드 솔루션, 데이터 이전 솔류션 등을 이용한 서비스 환경에 최적화된 구성으로 데이터 이전, 전자기록물 관리 사업 협업 모델 개발
- 전자기록물 장기보존 관련 신규 비즈니스 영역 창출에 활용

제6장 기타 중요 변경 사항

1. 인력 변경

○ 해당사항없음

2. 연구비 예산 변경

○ 해당사항없음

제7장 참고문헌

○ 해당사항없음

제8장 첨부 서류 목록

○ 해당사항없음

세부 연구과제 요약

과제 고유번호	자동부여		공개가능여부	공개
주관과제명	데이터세트 유형 전자기록의 장기보존기술 연구			
제 2 세부과제명	클라우드 기반 전자기록의 장기보존기술개발 테스트베드 구축 및 에뮬레이션 시험•검증			
연구책임자	성 명	조 철 용		
	소속 기관명	(주)이노그리드		
	전자우편	*****	전화번호	***-****-****

○ 연구목표

· 본 연구는 데이터세트 유형 전자기록의 장기보존을 위해 클라우드 기반 전자기록의 장기보존기술개발 테스트베드를 구축하고 데이터세트 유형, 규모, 환경 등에 따라 에뮬레이션 보존 방식을 시험하고 검증하는 것을 목표로함

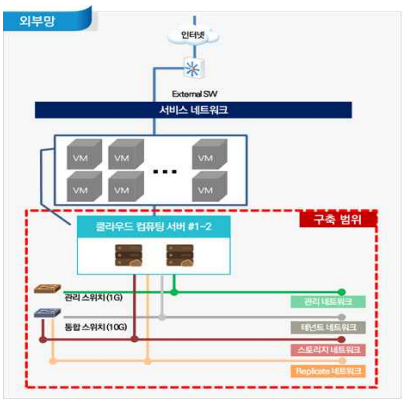
· 에뮬레이션 보존 방식에 따른 기술적합도 검증을 위한 클라우드 기반 전자기록의 장기보존기술개발 테스트베드 구축
  · 에뮬레이션 시험 검증을 위한 하드웨어 시스템 구축
  · 에뮬레이션 시험 검증 환경 제공을 위한 오픈스택 기반의 클라우드 인프라 환경 구축

· 데이터세트 유형 전자기록의 데이터세트 유형, 규모, 환경 등에 따른 에뮬레이션 보존 방식 시험 및 검증
  · 선정된 데이터세트의 에뮬레이션 후 원천 데이터세트와의 정합성 검증 항목 마련 및 점검
  · 에뮬레이션 사전 준비, 에뮬레이션 절차, 정합성 검증항목 등 도출

○ 연구내용

· 보존방식에 따른 기술적합도 검증을 위한 테스트베드 구축
  · 에뮬레이션 시험 검증을 위한 하드웨어 시스템 구축
    - 에뮬레이션 시험 검증 요구사항에 따라 클라우드 인프라 규모 사이징, 하드웨어 아키텍쳐 및 네트워크 구성(안) 수립
    - 기존 클라우드 인프라 구축 경험을 통해 안정성이 검증된 하드웨어 장비 확보
    - 참여기업들 간의 원활한 연구개발 추진을 위해 하드웨어 시스템을 민간 데이터센터에 구축
  · 오픈스택 기반 클라우드 인프라 환경 구축
    - 안정성이 검증된 오픈스택 퀸즈 버전으로 클라우드 인프라 환경 구축
    - 에뮬레이션 방식의 전자기록물 보존 시험·검증을 위해 가상머신, 도커/컨테이너 환경 제공 수준의 클라우드 인프라 환경 구축
    - 이를 위해, 다양한 오픈스택 프로젝트 중 불필요한 프로젝트를 제외한 최적의 프로젝트들만 설치

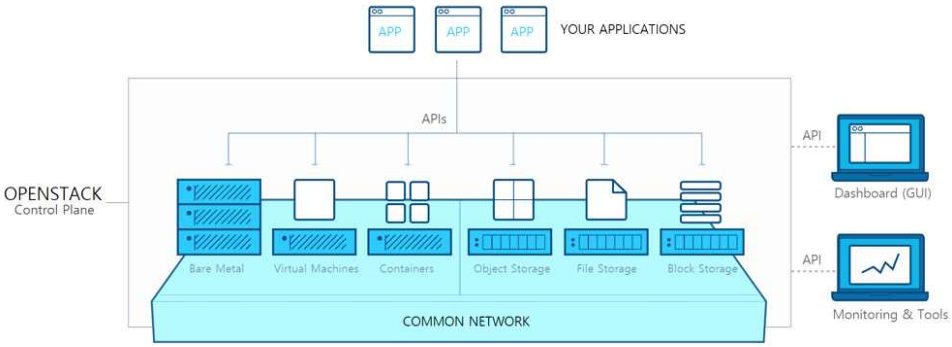
· 에뮬레이션 시험 검증을 위한 하드웨어 시스템 구축



예정 장비 스펙

구분	사양	수량	비고
클라우드 컴퓨팅 서버	• CPU: 24C/48T • RAM: 256GB • DISK1: SSD 480GB * 2EA • GlusterFS DISK: 1TB * 6EA	2대	GlusterFS
통합 스위치	• 10G 12Port Ethernet	1대	
관리 스위치	• 1G 8Port Ethernet	1대	

· 오픈스택(OpenStack)기반 클라우드 인프라 환경 구축



프로젝트명	서비스 내용	비고
glance	· 이미지 서비스	
horizon	· 포탈 서비스	
keystone	· 인증서비스	
neutron	· 네트워크 서비스	
nova	· 컴퓨트 서비스	
heat	· 오케스트레이션 서비스	
magnum	· docker 서비스	

오픈스택잇 UX 특징

- 클라우드잇 디자인 노하우에 기반한 UX(사용자 경험) 극대화
- 위자드를 통한 자원 생성 화면 제공
- 목록과 상세내용 동시 표현
- 화면 이동을 최소화한 구성
- 관리자/사용자 권한에 따른 서로 다른 기능 및 화면 구성 제공
- 사용자를 위한 기능을 별도로 제공하는 셀프 서비스 포털 제공

· 데이터세트 유형 전자기록의 에뮬레이션 시험

· 선정된 데이터세트의 에뮬레이션 보존 방식 테스트

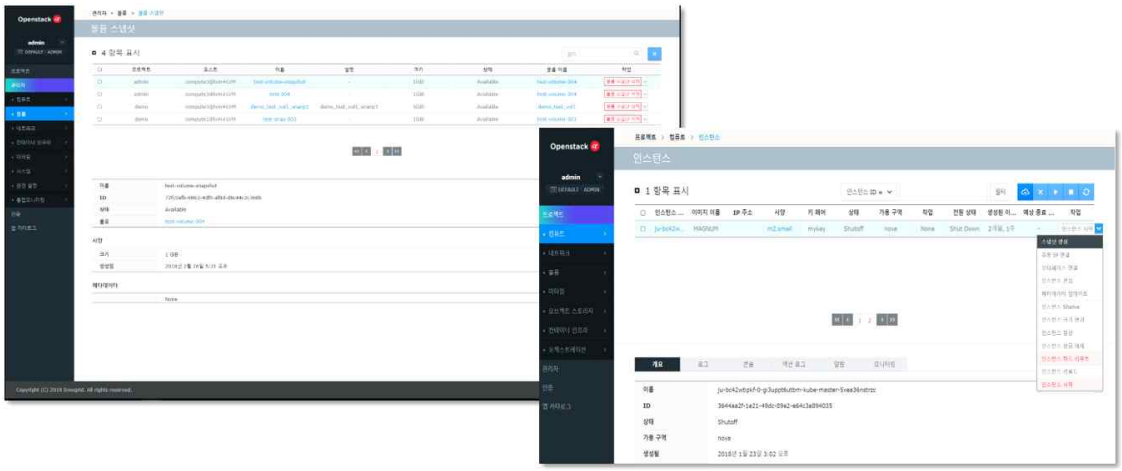
- 공공기관의 테스트용 데이터세트 선정을 위한 과제수행부서의 기관 방문, 면담 등을 지원하고, 선정된 데이터세트 운영 현황 분석을 통해 보존 범위(OS, 데이터셋, 애플리케이션 등)를 정의

- 클라우드 테스트베드 기반 가상화 환경에 데이터세트 및 운영 환경을 마이그레이션하고, 클라우드의 스냅샷, 백업 기능을 통해 가상화 이미지로 보존

- 보존된 가상화 이미지로부터 복원 후 데이터 세트 정합성 검증

· 선정된 데이터세트의 에뮬레이션 후 원천 데이터세트와의 정합성 검증항목 마련 및 점검

· 선정된 데이터세트의 에뮬레이션 보존 방식 테스트



스냅샷 기능

- 인스턴스, GPU 인스턴스, 볼륨 데이터의 시점 백업 기능인 스냅샷 기능 제공

볼륨 백업 기능

- Full Backup
- Incremental Back up (증분 백업)

· 선정된 데이터세트의 에뮬레이션 후 원천 데이터세트와의 정합성 검증항목 마련 및 점검

· 데이터세트 유형, 환경 등 특성에 따라 에뮬레이션 보존 방식 적합 모델 도출

· 데이터세트 유형, 규모, 시스템 환경에 따라 에뮬레이션 전후 정합성 검증 항목 도출

○ 연구성과(응용분야 및 활용범위포함)

· 응용 분야
  · 공공 전자기록물 저장 및 관리 서비스
  · 민간 기업 데이터 저장 및 관리 서비스
  · 데이터 비즈니스 마케팅 산업의 데이터 저장 관리 서비스

· 활용 범위
  · 공공기관의 공공 데이터, 전자기록물 보관 및 관리
  · 민간기업의 사원, 계약 정보 등 비즈니스 데이터 관리
  · 데이터 비즈니스 기업의 생산, 재생산 데이터 관리
  · 다양한 환경에서의 전자기록물 데이터 포맷 및 애플리케이션 테스트

○ 참여연구원

성 명	소속/직위	성 명	소속/직위
조철용	㈜ 이노그리드 / 실장	윤은선	㈜ 이노그리드 / 수석연구원
김선홍	㈜ 이노그리드 / 선임연구원	정혜원	㈜ 이노그리드 / 부장
서동민	㈜ 이노그리드 / 과장	조종문	㈜ 이노그리드 / 부장
우상수	㈜ 이노그리드 / 책임연구원		

Keywords (5개 내외)	*한글*	전자기록물, 장기보존기술, 에뮬레이션, 클라우드
	영문	electric records, long-term preservation technology, emulation, cloud

데이터세트 유형 전자기록 장기보존

첨 부

첨부 목차

[첨부01] 2019 ICLIS 대만학회 발표자료 1 ····· 1P

[첨부02] 2019 ICLIS 대만학회 발표자료 2 ····· 2P

[첨부03] 한국기록관리학회 논문지 19권 4호 2019년 11월 게재(KCI) ····· 3P

[첨부04] 용어정리 ····· 36P

[첨부01] 2019 ICLIS 대만학회 발표자료 1

첨부01	2019 ICLIS 대만학회 포스터

A Study on Administrative Information Datasets as Evidence of Public Service

JungEun Lee[1+], EunHa Youn[2+], Geon Kim[3+]

1. lepina@naver.com, 2. eunhayoun@gmail.com, 3. godardkim@jbnu.ac.kr
+ Graduate School of Archives and Records Management, Chonbuk National University, Jeonju, Republic of Korea
Institute of Culture Convergence Archiving, Chonbuk National University, Jeonju, Republic of Korea

CHONBUK NATIONAL UNIVERSITY

Introduction

As many administrative information systems were established through the promotion of e-government, new forms of records such as web, e-mail, and datasets emerged beyond electronic records. One of the fastest-growing types of records is in the datasets. In SOUTH KOREA , there are about 17,000 types of systems used by government agencies. Administrative information datasets are not only records of evidence value, but also high information value. Therefore, it is necessary to establish datasets archival system so that they can be systematically acquired, managed, preserved, and effectively utilized. Recently in Korea, work is underway to treat administrative datasets as a form of electronic records and archive them. For records to be recognized as evidence, four attributes must be maintained: authenticity, reliability, integrity, and availability. Therefore, in this study, we have identified considerations for managing administrative information datasets by archiving them.

Understanding Administrative Information Datasets as Records

The key to managing records of administrative information data sets is to secure business activities and linkage information and to ensure the ' **FixAbility**' of records. Records are not merely aggregates of data, they are the product or outcome of a characteristic event, and the record must be in a fixed form for the record to have value as evidence. However, it is not necessary to manage all the information in the administrative system as a record, and it is necessary to select and record the information that is evidence of the work through business process analysis.

Cnosideration for Archiving Administrative Information Datasets

Selection of Datasets

Not all datasets that present in the administrative information system should be managed as records. Therefore, screening questions about which datasets to archives should take precedence. Key to the screening requirements for datasets is to derive evidence and evidence requirements to be retained, and to identify the content or data that constitutes the evidence.

Acquisition of Datasets

The acquisition is the act of securing records with an electronic recording system just at the time the records are produced. Securing records at the time of production is the first step in record management. Because of this, obtaining datasets is very important. For records to be recognized as evidence, these four attributes must be maintained: authenticity, reliability, integrity, and availability. Therefore, datasets should be reclaimed by considering the requirements to maintain the above four attributes as a type of record.

Depresentation of appearance

Reproducibility of electronic records can be understood as an attribute that should be artificially provided through the monitor screen so that people can read and interpret not only the contents of the electronic records but also the appearance and functions at the time of use. Datasets are not reproducible cannot be accepted as authentic electronic records because they do not meet the availability requirements.

Long-term preservation of Datasets

The purpose of archiving datasets is to use. Therefore, establishing long-term conservation strategies for use is important. Long-term conservation strategies should ensure access to and utilization of records. Securing preserved metadata is essential for archiving datasets of public information. Datasets should be preserved with information describing the entire process from generation to management and preservation of records. Preserved metadata management should cover more diverse and detailed items than other metadata to maintain long-term viability, re-endurance, understandability, authenticity, and integrity.

CONCLUSION

Conclusion

Administrative information datasets, like other records, should be identified and acquired from the production stage, and a standardized record management system should be established so that they can be selected for long-term preservation and future use. A lot of digital information has been accumulated in the administrative information system, which can be seen as an important resource for the nation. To this end, some considerations are given for archiving datasets that have very different characteristics than traditional records.

CHONBUK NATIONAL UNIVERSITY

This work was supported by the Ministry of Education of the Republic of Korea and the National Research Foundation of Korea (NRF-2016S1A5B8913575)
This research was supported by a fund from the Archive Management Research Program of the National Archives, Korea (Study on long-term preservation technology of dataset-type electronic records, 2019)

[첨부02] 2019 ICLIS 대만학회 발표자료 2

CHONBUK NATIONAL UNIVERSITY

A Study on the Long-Term Preservation of Digital Information Resources

Hui Jeong Han [1], Dong-min Yang [2]

[1] Institute of Culture Convergence Archiving, Chonbuk National University, Jeonju, Republic of Korea (freebirdhhj@naver.com)

[2] Graduate School of Archives and Records Management, Chonbuk National University, Jeonju, Republic of Korea,
Institute of Culture Convergence Archiving, Chonbuk National University, Jeonju, Republic of Korea (dmyang@jbnu.ac.kr)

Introduction

As data becomes a key raw material for the future industry, there is a growing interest in management and utilization of various information resources produced and shared digitally. In the era of the 4th Industrial Revolution, it is very important to systematically preserve and manage digital information resources as a core asset of national competitiveness. However, there are many difficulties in preserving and managing such digital information resources. Types and quantities of digital information resources are increasing exponentially, and the environment and technologies are changing. As a result, threats to long-term preservation of digital information resources are increasing. In order to overcome those threats, it is necessary to develop **new policies and strategies for long-term preservation of digital information resources**, which will make digital information resources accessible and available even after a long period of time.

Polices for Long-Term Preservation of Digital Information Resources

The purpose of long-term preservation of electronic records is to access to them, to maintain their authenticity and to server as evidence even after a long period of time by eliminating the threats against long-term preservation. The basic principles of long-term preservation policy are as follows.

First, preservation policies should maintain sustainability which will not be affected for a long period of time by environmental changes. That is, by introducing sustainable preservation policies, strategies, and technologies, electronic records must be managed to ensure authenticity, integrity, reliability and availability in all processes from production to utilization.

Second, preservation policies should be flexible enough to cope with changes of technology. To preserve electronic records safely and efficiently for a long period of time, we need to examine rapidly developing digital technologies and have skills, people, and organization that can improve sustainability and efficiency.

Third, preservation policies should be scalable to facilitate expansion and reduction of computing resources and functions. Even if electronic records are rapidly increasing or shrinking, system design and construction should be executed to accommodate the computing resources without massive redesign or re-installation of computing resources.

Fourth, preservation policies should comply with both domestic and international standards. To ensure the transparent and systematic long-term preservation of electronic records, every records management procedure must be conducted through standardized management policies and records management standards based on laws and ordinances.

Current status of adoption of document preservation format

Target : US NARA, Canada LAC, Australia NAA

- Good : 2+ or more of the preferred formats
- Fair : 1 of the preferred formats or 2+of the allowed formats
- Poor : 1 of the permitted formats

type	Good	fair	poor
Document(text)	PDF/A-1, PDF/A-2, TXT	EPUB, ODF, ODT, PDF, DOCX, DOC	EPUB, OOXML
Presentation	PDF/A-1, ODP	PDF/A-2, PPT, PPTX	
Dataset	ASCII, CSV	JSON, XML, ODS, XLS, XLSX, EBCDIC	DBF, Unicode, SIARD, MS Access
Still image	TIFF, JP2, PNG	ODG(OTG), PDF/A, PDF/A-1, JPG, PDF/A-2, SVG, DNG, GIF, JPG	DICOM, Exif
Audio	BWF	FLAC, WAV/WAVE, AIFF, MP3	AAC, MPEG4
Video	DPX	DCDM, AVI, MXF, MOV, DCP, WMA, MPG	
Web	WARC	ARC, XHTML	HTML
E-mail	EML, MBOX	PST, MSG	XML
CAD		STEP, X3D, DXF, AutoDesk's Drawing File, PDF/E	U3D, PRC, DWG
GIS	GML, GTIFF, KML	ESRI SHP, TIGER, BIL, BIP, BSQ, DEM, ESRI Arc/Info ASCII Grid, ESRI ArcInfo Export (E00), TerraGo Geospatial PDF	Vector Product Format, SDTS, CCOGIF, DIG3, IHO S-57

Strategy for Long-Term Preservation of Electronic Records

Overview of major archive institution criteria

Criteria	Canada (LAC)	American Library of Congress (LOC)	the United Kingdom (TNA)	the United States (NARA)	National Archives of Korea
Openness / Transparency	○	○	○	○	○
Adoption	○	○	○	○	○
Stability/Compatibility	○		○		○
Dependency / Interoperability	○	○	○		○
Standardization	○				
Compression				○	
Impact of Patent		○	○		
Technology Protection Mechanism		○		○	
Metadata Support		○	○	○	○
Authenticity					○
Expressiveness					○
Search Function					○

Evaluation method according to document preservation format selection criteria

$P_{total} = P_O + P_L + P_C + P_{ID} + P_V + P_{IO} + P_D + P_R$

- Suitability : $21 \le P_{total} \le 25$
- partial suitability : $15 \le P_{total} \le 20$
- unsuitability : $0 \le P_{total} \le 14$

criteria	Detailed criteria	good	fair	poor
Open Disclosure	(P_O) Open to the public	4	3	1
	(P_L) License	3	2	1
	(P_C) Standardization organization or a group	3	2	1
Stability	(P_{ID}) Identifiability	3	2	1
	(P_V) Ease of verification	3	2	1
(P_{IO}) Interoperability		3	2	1
(P_D) Self-Documentation		3	2	1
(P_R) Retrieval		3	2	1
Total		25	17	8

Example of evaluation of document preservation format : SIARD

criteria	contents	evaluation	score
Open Disclosure	• open to the public : openness - Swiss e-Government standards (eCH) : https://www.ech.ch/standards/38716 - SIARD suite software : https://github.com/sfa-siard/SiardGui/releases	Good	4
	• license : Open Source (CDDL-License) on Github	Good	3
	• standardization organization of group : Swiss e-Government standard (eCH-0165)	Fair	2
Stability	• identifiability : XML, SQL - As a ZIP package, DB structure and data are disclosed in the SIARD standard.	Good	3
	• ease of verification : SIARD can be verified through reference program of SIARD SUITE and GUI	Good	3
Interoperability	• There is a reference program written with JAVA (ex : SIARD GUI, SIARD SUITE)	Good	3
Self - Documentation	• Metadata - Including a schema that represents the structural metadata of the relational database	Good	3
retrieval	• it is an XML-based text file that can be retrieved	Good	3
total		suitability	24

This work was supported by the Ministry of Education of the Republic of Korea and the National Research Foundation of Korea (NRF-2016S1A5B8913575).
This research was supported by a fund from the Archive Management Research Program of the National Archives, Korea (Study on long-term preservation technology of dataset-type electronic records, 2019).

[첨부03] 한국기록관리학회 논문지 19권 4호 2019년 11월 게재(KCI)

첨부03-2 | 논문

한국기록관리학회지
19(4), 1-33, 2019.11
http://dx.doi.org/10.14404/JKSARM.2019.19.4.001
pISSN 1598-1487 eISSN 2671-7247

JKSARM
Journal of Korean Society of Archives and Records Management

클라우드 컴퓨팅 기반 에뮬레이션 전략을 활용한 전자기록 장기보존 방안 연구

A Study on Long-Term Electronic Records Preservation Using Cloud-Based Emulation Strategy

이봉환(Bong-Hwan Lee)[1], 한희정(Hui-Jeong Han)[2], 조철용(Cheolyong Jo)[3], 왕호성(Ho-sung Wang)[4], 양동민(Dongmin Yang)[5]

E-mail: blee@dju.kr, freebirdhhj@naver.com, cheolyong@innogrid.com, kinghosung@gmail.com, dmyang@jbnu.ac.kr

[1] 제1저자 대전대학교 전자·정보통신공학과 교수
[2] 전북대학교 문화융복합아카이빙연구소 전임연구원
[3] ㈜이노그리드 실장
[4] 국가기록원 기록연구사
[5] 교신저자 전북대학교 기록관리학과 부교수, 문화융복합아카이빙연구소 연구원

OPEN ACCESS

논문접수 2019.7.29
최초심사 2019.8.4
게재확정 2019.11.10

ORCID
Bong-Hwan Lee
https://orcid.org/0000-0003-0088-0530
Hui-Jeong Han
https://orcid.org/0000-0003-2153-5963
Cheolyong Jo
https://orcid.org/0000-0003-1167-4695
Ho-sung Wang
https://orcid.org/0000-0002-1955-7998
Dongmin Yang
https://orcid.org/0000-0002-4029-9372

© 한국기록관리학회
This is an Open Access article distributed under the terms of the Creative Commons Attribution-NonCommercial-NoDerivatives 4.0 (https://creativecommons.org/licenses/by-nc-nd/4.0/) which permits use, distribution and reproduction in any medium, provided that the article is properly cited, the use is non-commercial and no modifications or adaptations are made.

▪본 연구는 "2019년 행정안전부 국가기록원 기록관리 연구개발사업"의 연구비를 지원받아 수행되었음.
▪이 논문은 2018학년도 대전대학교 교내 학술연구비 지원에 의해 연구되었음.

http://ras.jams.or.kr

초 록

전자기록 장기보존의 핵심은 전자기록이 최초로 생성되고 활용되었던 본래의 기능적 속성과 모습 등을 담고 있는 비트스트림을 오랫동안 그대로 유지하는 것이다. 대부분 국내외 주요 아카이브 기관들은 경제적, 기술적인 측면을 고려하여 원본의 비트스트림을 담고 있는 포맷의 안정성, 신뢰성에 따라 새로운 포맷으로의 변환을 허용하는 마이그레이션을 주전략으로 채택하고 있다. 그러나 전자기록 유형이 다양해지고 범위가 확장됨에 따라 마이그레이션 단일 전략은 다양한 문제점들을 발생시키고 있다. 그러므로 본 연구에서는 클라우드 컴퓨팅 기술을 활용하여 전자기록의 본래의 기능적 속성과 모습을 담고 있는 비트스트림을 변경하지 않으면서, 전자기록이 생산·활용된 시스템 및 응용 환경까지 재생산하는 에뮬레이션 전략의 적용 가능성을 검토하고자 한다.

ABSTRACT

The key to the long-term preservation of electronic records is the long-term retention of the bitstream that contains the original functional attributes and appearances of the electronic records that were originally created and utilized. Considering economic and technical aspects, most of the major domestic and foreign archives have adopted a migration strategy that allows conversion to a new format depending on the stability and reliability of the format containing the original bitstream. However, as the type of electronic records varies and its scope continuously expands, several problems arise if a single migration strategy is observed. Therefore, the applicability of the emulation strategy is examined in this paper to reproduce the system and application environment where electronic records are created and utilized. As such, electronic records can be maintained without any changes in the bitstream by utilizing cloud computing technology.

Keywords: 장기보존, 전자기록, 클라우드 컴퓨팅, 에뮬레이션, 마이그레이션
Long-term Preservation, Electronic Records, Cloud Computing, Emulation, Migration

1. 서 론

1.1 연구배경 및 목적

디지털 기술의 발전으로 전자기록의 활용도가 급속도로 증가하고 있으며, 이에 따라 기존의 종이기록에서 전자기록 관리체계로 전환되어가고 있다. 디지털 객체로 생성되고 활용되는 전자기록은 종이기록과 달리 객체의 비트스트림이 조금이라도 훼손되면 전자기록을 확인할 수 없을 수도 있다. 현재 전자기록의 생산 및 활용을 위한 관련 기술들은 많이 있으며, 그 기술들이 하드웨어에서 응용 소프트웨어까지 서로 조직적이고 복잡하게 연결되어 있기 때문에 하나의 연결고리라도 끊기거나 이상이 생기면 전자기록을 사용할 수 없게 된다. 또한 전자기록 관련 기술들이 빠르게 발전함에 따라 새로운 기술들도 계속 등장하고 있어 오랜 시간이 지난 후에는 전자기록을 확인할 수 없는 상황이 발생할 수 있다. 이렇듯 기록 자체를 확인할 수 없는 위험한 상황 때문에 전자기록의 장기보존은 종이기록에서 전자기록으로 전환되는 시점에서 해결되어야 하는 가장 중요한 문제이다. 이를 해결하기 위해 아카이브 기관들은 전자기록 장기보존을 위한 다양한 전략을 수립하고 있다.

국내외 주요 아카이브 기관들은 전자기록 장기보존을 위한 주전략으로 마이그레이션(Migration)을 채택하고 있다. 마이그레이션 전략을 통해 대부분의 전자기록에 대한 일괄적인 보존전략을 세울 수 있다. 일반적으로 마이그레이션을 수행하는 디지털 객체의 포맷 변환 기술을 해당 포맷을 개발한 기업에서 기본적으로 제공하거나 오픈 소스 프로젝트로 무료로 배포되어 왔다. 따라서 마이그레이션 전략은 상대적으로 기술적인 장벽이 낮고 비용이 적다는 이유로 많이 채택되어 왔다. 그러나 마이그레이션 방식은 디지털 객체의 포맷을 더 안정적인 포맷으로 변환하는 방식이기 때문에 비트스트림이 변경될 수 밖에 없고 이로 인해 비트스트림의 손실이 발생한다는 본질적인 단점을 갖고 있다. 전자기록의 유형이 전자문서에서 시청각기록물, 웹기록물, 행정정보데이터세트 등 다양한 형태로 그 범위가 확장되고 있으며, 디지털 객체는 물론 해당 객체를 구동하는 시스템 하드웨어 및 소프트웨어를 포함한 운영 환경까지 보존하는 것을 고려해야 하는 상황들이 생기고 있다. 이러한 상황을 해결하기 위해 장기보존 전략 중 하나인 에뮬레이션(Emulation) 방안이 대두되고 있다. 에뮬레이션 전략은 소프트웨어 기능 및 디지털 객체가 생성되고 활용되었던 예전의 모습을 그대로 현재의 컴퓨팅 환경에서 재현하기 위해 당시 컴퓨팅 운영 환경을 재생산하는 것을 의미한다. 에뮬레이션 전략은 기록물의 생산 당시의 모습 그대로 보존하고 재현할 수 있다는 점에서 기록관리 분야에서 상당한 관심을 받고 있으며 마이그레이션 전략과 함께 지속적으로 논의되고 있다. 다만, 에뮬레이션 전략을 구현하기 위한 경제적인 비용이 높고, 여전히 기술적으로 해결해야 할 난제들이 남아 있기 때문에, 여러 차례 검토 및 연구들이 진행되어 왔음에도 불구하고 적극적으로 활용되지는 못하고 있다.

최근에 기업뿐 아니라 중앙정부부처와 공공기관들까지 자체적으로 서버들을 운영하지 않고, 처음부터 클라우드 컴퓨팅(Cloud Computing)

http://ras.jams.or.kr http://dx.doi.org/10.14404/JKSARM.2019.19.4.001

환경에 서버들을 구축하거나, 기존의 서비스를 제공하고 있던 시스템들을 클라우드 컴퓨팅 환경으로 전환하는 속도가 가속화되고 있다. 클라우드 컴퓨팅 환경에서 사용되고 있는 가상화(Virtualization) 기술은 에뮬레이션의 핵심 기술로 실제 클라우드 전환 사업에 활발히 도입되는 단계까지 올 정도로 성숙된 기술이다. 그러므로 클라우드 컴퓨팅 기술은 장기보존 전략에 에뮬레이션을 적용하는데 걸림돌이 되어 온 여러 가지 문제들을 해결하면서 전자기록을 장기보존하는데 활용될 수 있다.

클라우드 컴퓨팅 기술은 가상화 기술을 기반으로 일반 사용자 또는 기업에게 CPU, 메모리, 디스크 등의 '인프라', 소프트웨어 개발 환경인 '플랫폼', 업무 수행을 위한 '소프트웨어'를 본인의 PC에 설치하지 않고 인터넷 연결을 통해서 서비스를 제공하게 하는 기술이다. 데이터센터(Data Center)와 같은 대규모 비용이 들어가는 기반시설을 직접 설치·유지 관리하지 않고, 필요한 만큼만 컴퓨팅 자원을 사용할 수 있으며, 서버 구매, 유지보수 비용을 크게 절감할 수 있다는 믿음 때문에 연구 개발되어 왔다.

클라우드 컴퓨팅은 기존 시스템 전환 및 신규 시스템 구축 관점 이외에 장기보존 관점에서도 활용가치가 높은 기술이다. 원하는 사양의 컴퓨팅 자원을 유연하게 제공해 줄 수 있는 클라우드 컴퓨팅의 가상화 기술과 분산처리 기술은 전자기록이 생산·활용되는 시스템 운영 환경을 보존하는 방식인 에뮬레이션 전략에 대한 해결책을 제시할 수 있다. 클라우드 컴퓨팅 기술과 서버 백업 시 사용되는 도구(예, ZConverter, virt-p2v 등)와 결합된다면 전자기록이 생산되었던 시스템과 유사한 운영 환경을 만드는 수준을 넘어 완전히 동일한 환경을 제공할 수 있기 때문이다.

에뮬레이션은 본래의 기능과 모습을 재현한다는 점에서 기록관리 측면에서 가장 최적화된 방식으로 주목받았지만 경제적인 비용과 기술적인 한계로 도입이 어려웠다. 클라우드 컴퓨팅은 에뮬레이션 방식을 장기보존 전략에 도입하는데 있어 소용되는 경제적인 비용을 낮추고 기술적인 한계를 극복할 수 있는 방안을 제시할 수 있다. 다양한 유형의 전자기록들을 보존하는데 있어 마이그레이션 단일 전략으로는 한계가 드러나고 있는 상황에서 클라우드 컴퓨팅 기술을 기반으로 하는 에뮬레이션 전략의 타당성을 적극적으로 검토하는 것이 필요하다.

이에 본 논문에서는 클라우드 컴퓨팅 기반 에뮬레이션 전략 방안을 연구하여 이를 활용한 전자기록 장기보존 방안을 제시하고자 한다. 또한, 실제 에뮬레이션을 수행할 수 있는 오픈스택(OpenStack)을 활용하여 테스트베드를 구축하고 에뮬레이션 기법을 실험하여 클라우드 컴퓨팅 기반 에물레이션 전략의 실제 적용 가능성을 타진하고자 한다.

1.2 연구범위 및 방법

본 연구에서는 클라우드 컴퓨팅 기반 에뮬레이션 전략을 활용한 전자기록 장기보존 방안을 제시하기 위해 가장 먼저 국내외 클라우드 컴퓨팅 기술을 조사 및 분석한다. 조사 및 분석 결과를 바탕으로 가장 현실적이면서 전자기록 장기보존에 적합한 형태의 테스트베드를 구축한다(〈표 1〉 참고). 다양한 전자기록 장기보존을 위한 에뮬레이션 테스트 및 분석을 통해 다

〈표 1〉 클라우드 컴퓨팅 테스트베드 구성

구분		내용
하드웨어 구성	컴퓨팅 서버	2대(CPU:Intel Xeon, 12Core, 2.1GHz/85W)
	스위치	2대(통합스위치:10G 12Port, 관리스위치:1G 8Port)
소프트웨어 구성	클라우드 구성	오픈스택(OpenStack) Queens
	가상화	KVM(Kernel-based Virtual Machine)
	에뮬레이터	QEMU(Quick Emulator)
	클라우드 관리	오픈스택잇(OpenStackit)
	시스템(템플릿) 이미지 변환 도구	ZConverter, virt-p2v

양한 전자기록 유형에 대해 에뮬레이션 기반 장기보존 방안을 제시한다.

1.3 선행연구 및 사례조사

국내 전자기록 클라우드 컴퓨팅 기술을 전자기록관리에 활용하기 위한 연구는 상당히 활발하게 진행되어 왔다. 그러나 대부분의 연구는 기존의 전자기록관리시스템을 클라우드 컴퓨팅 기반 환경으로 변환하는 내용에 초점이 맞춰져 있다. 이승억, 설문원(2017)에서는 기록관리의 환경이 변화함에 따라 우리나라 전자기록관리가 전면적으로 재설계되어야 함을 주장하면서 향후 다루어야 할 정책 영역과 그 방향성을 제시하였다. 새로운 환경변화에 대처함에 있어 클라우드 컴퓨팅은 전자기록관리의 핵심 플랫폼으로 간주되고 있다. 임지훈, 김은충, 방기영, 이유진, 김용(2014)는 대량으로 생산되는 전자기록을 관리하기 위해 비용대비 효과가 뛰어난 클라우드 컴퓨팅을 전자기록관리시스템에 도입하는 방안을 제시했다. 또한, 해외 아카이브 기관의 클라우드 컴퓨팅 활용 사례 및 클라우드 컴퓨팅 환경에 대한 기록물 관리 지침서 등을 기반으로 클라우드 컴퓨팅 도입 방안에 대해 구체적으로는 구성하고, 현재 전자기록관리시스템의 현황을 분석하여 네 가지 문제점을 도출했다. 이를 고려하여 기록관 업무분야와 기존 시스템의 구성요소를 중심으로 클라우드 컴퓨팅 환경 모델 적용 방안과 전자기록관리시스템 보안 프로세스를 제시하였다. 정예용, 심갑용, 김용(2014)는 전자기록관리시스템 중에서 중앙기록물관리시스템인 영구기록관리시스템의 보존 및 활용 방식을 제안하였다. 영구기록물관리기관의 업무, 자료, 시스템의 특성을 파악하고 이를 기반으로 하는 클라우드 컴퓨팅 기반의 영구기록물관리시스템의 보존 및 활용 모델을 도출하였으며, 하드웨어 및 소프트웨어 자원을 클라우드 컴퓨팅 모델별로 매핑하고 업무 프로세스를 클라우드 컴퓨팅 기반으로 전환하는 방안을 제시하였다. 김기정, 신동수(2018)도 영구기록물관리시스템을 클라우드 컴퓨팅 환경으로 전환하는 방안을 제시하였다. 이 연구는 2017년 국가기록원 연구개발 사업 '차세대 기록관리 모델 재설계 연구'의 일환으로 수행된 연구로 국가기록원 영구기록물관리시스템에 대한 구체적인 조사 및 분석이 이루어졌으며 이를 기반으로 클라우드 환경으로 전환하기 위한 단계적인 방안을 제시하였다. 김주영, 김순희(2019)는 전자기록생산시스템인 업무관리시스템 또는 온-나라 시스템에서 기록관

리시스템으로 전자기록물을 이관할 때 발생하고 있는 디지털 객체의 무결성 훼손 문제를 해결하기 위해 클라우드 저장소를 활용한 논리적 전자기록 이관 방안을 제안하였다.

기존 전자기록관리 분야의 클라우드 컴퓨팅 기술 활용 연구는 기존 각 기관들이 보유하고 있는 다양한 전자기록관리의 시스템들을 안정적으로 클라우드 컴퓨팅 환경으로 전환하여 비용과 관리의 효율성을 제고하는 방안을 제시하는데 그치고 있다. 이는 우리나라가 전자기록을 생산·활용·관리·이관·보존하는 온-나라 업무관리시스템과 RMS(Records Management System)를 클라우드 컴퓨팅 기술 기반으로 통합하는 것을 추진하고 있기 때문에 기존의 국내 연구들 역시 클라우드 컴퓨팅 환경으로 전환하기 위한 시스템 통합에 관한 연구가 주로 이루어진 것으로 보인다.

한편, 에뮬레이션 전략에 관한 국내 연구는 다음과 같다. 먼저 김명훈, 오명진, 이재홍, 임진희(2013)는 장기보존 전략으로서 에뮬레이션이 지닌 장단점을 도출한 다음, 에뮬레이션 관련 선진 사례인 CAMiLEON, KB, Planets, KEEP 프로젝트를 조사, 분석하였다. 이러한 분석을 기반으로 향후 우리나라 전자기록에 대한 시사점 및 적용 방향을 제시하였다. 국가기록원(2013)은 한국형 에뮬레이션 전략 방안을 제시하였으며, Oracle에서 오픈 소스로 배포하고 있는 VirtualBox라는 VM(Virtual Machine) 에뮬레이터를 기반으로 한국형 에뮬레이션 프로토타입을 개발하여 전자기록 장기보존 전략으로서의 에뮬레이션 방식의 적용 가능성을 점검하였다. 이들 연구는 국내외 에뮬레이션 사례를 구체적으로 조사 및 분석하였고, 에뮬레이션 전략을 타당성 점검을 위해 상용 에뮬레이터를 활용하여 타당성을 점검하였다는 점에서 의미를 갖는다. 그러나 전자기록의 에뮬레이션 적용 가능성을 검토하고 있지만 기술적인 어려움과 높은 경제적인 비용으로 현재로서는 적용 가능성에 대해 의문을 남기고 있다. 반면, 왕호성, 설문원(2017)은 행정정보데이터세트 기록의 재현과 관련하여 에뮬레이션에 주목하였다. 즉, 에뮬레이션 기법은 가상화, 클라우드 서비스, 원격데스크톱 프로토콜(RDP: Remote Desktop Protocol)을 결합하여 구현될 수 있으며, 이를 서비스로서의 에뮬레이션 전략(EaaS: Emulation as a Service)으로 제안하였다. 또한, 전자기록관리 관련된 문제들을 정보통신 분야와의 협력을 통해서 극복할 수 있음을 언급하고 있다.

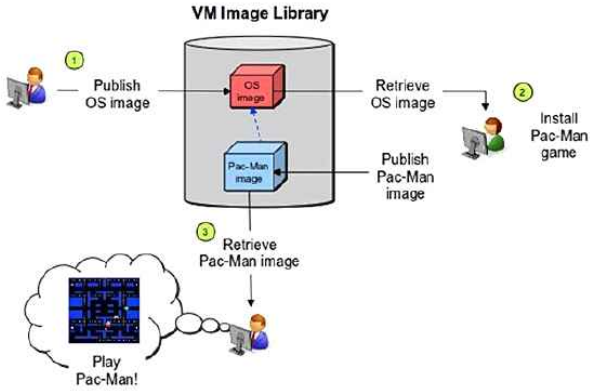
선행연구를 종합해 보면, 에뮬레이션은 다양한 측면에서 전자기록 장기보존 전략으로서의 장점이 있지만, 주전략으로 도입하기에는 경제적인 비용과 기술적인 한계 때문에 실현가능성이 낮다고 여겨졌다. 그러나 상용화 수준으로 기술이 성숙되어 가격 경쟁력을 가지고 있는 가상화, 클라우드 컴퓨팅, 원격접속 기술을 활용한다면 에뮬레이션은 또 하나의 장기보존 전략으로 고려될 수 있을 것으로 판단된다. 이에 본 연구에서 제안하는 클라우드 컴퓨팅 기반 에뮬레이션 전략을 활용한 전자기록 장기보존 방안 연구는 시의적절하다고 판단된다.

미국 카네기 멜론 대학교와 IBM연구소가 공동으로 추진하고 있는 올리브(OLIVE)[1] 프로젝트가 대표적인 클라우드 컴퓨팅 기반 에뮬레이션 기법이다(OLIVE, 2019). 올리브는 Windows3.1이나 초기 맥(MAC)컴퓨터의 환경을 가상화

기술로 이용하여 응용프로그램, 디지털 문서 파일, 운영 체제까지 포함한 "가상화 머신 이미지 또는 템플릿 이미지"들을 만들고 클라우드 컴퓨팅 환경 내부에 설치하여 누구나 외부에서 클라우드 컴퓨팅 환경에 접속하여 디지털 문서 파일을 원격으로 열람할 수 있는 서비스를 제공한다.

〈그림 1〉은 올리브 시스템 내에서 어떻게 콘텐츠들(OS, 응용SW 등)이 추가되고 운용되는지를 보여준다. 클라우드 컴퓨팅 기술로 구현된 "VM Image Library"에 사용자들이 OS와 해당 OS에서 동작하는 응용SW(Pac-Man)를 설치 및 업로드하면, 또 다른 사용자는 응용SW가 탑재된 OS를 클라우드 컴퓨팅 환경에 설치하여 응용SW를 활용할 수 있는 구조를 통해 서비스를 제공한다.

또 다른 사례로 독일의 bwFLA(Baden-Wuerttemberg Functional Long-term Archiving and Access) 프로젝트를 들 수 있다(bwFLA, 2019). 독일의 bwFLA는 클라우드 컴퓨팅 환경 기반의 에뮬레이션 기법을 구현하여 에뮬레이션 전략을 서비스(EaaS: Emulation As A Service)로서 제공하는 것을 목표로 하고 있다. 즉, 복잡한 디지털 기록이 지닌 특성을 장기보존 할 수 있으며 본래의 기능 및 모습을 경험하기 위한 가장 좋은 방법은 생성 당시의 응용 프로그램을 사용하는 것이라는 명제에서 출발하여 HW와 OS의 가상화 및 클라우드 컴퓨팅을 통해 구형 응용 프로그램을 실행시킬 수 있으며 일반인에게도 쉽게 접근할 수 있도록 구현하였다. MAAS/Juju, OpenStack로 클라우드 컴퓨팅 환경을 구축하였고, PPC, m68k, Intel-based



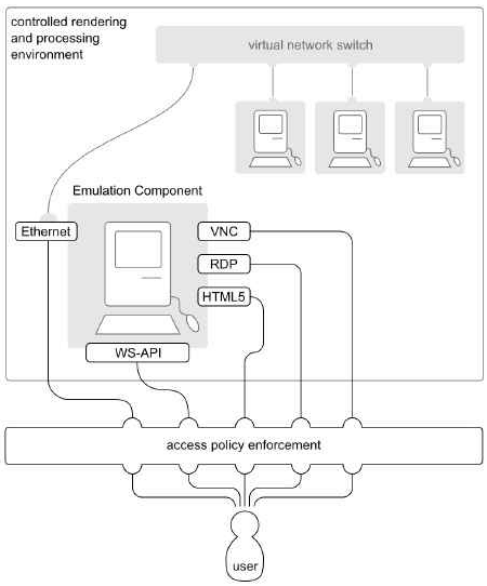
〈그림 1〉 올리브의 기술 구현 매커니즘

* 출처: Olive Archive, https://olivearchive.org/about/

1) OLIVE는 Open Library of Images for Virtualized Execution의 약자로 가상화 실행을 위한 템플릿 이미지를 제공하는 열린 도서관을 의미함.

x86등의 주요 CPU 구조와 OS/2, MS Windows, MacOS 등 주요 OS를 지원한다. 또한, 〈그림 2〉처럼 사용자에게는 VNC,[2] RDP, HTML5, WS-API,[3] Ethernet 등을 통해 인터넷으로 접근 서비스를 제공한다. 또한, 미국의 스탠포드 대학에서 약 40여 명의 디지털 아키비스트, 프로그래머, IT엔지니어, 학자, 기록관리 실무자들이 공동으로 출범시킨 BDAX(Born Digital Archiving & eXchange) 프로젝트는 2016년도 미국 아키비스트 협회(SAA) 연례회의에서 독일의 bwFLA의 에뮬레이션 기법을 실험하겠다는 계획을 밝혔다(왕호성, 설문원, 2017).

국외에서는 클라우드 컴퓨팅 환경에서의 에뮬레이션 서비스를 구축하고 제공하기 위한 연구 및 개발을 통해 상당한 결실을 맺고 있다. 그러므로 클라우드 컴퓨팅 기술이 성숙되어 있는 현재 시점에서 국내에서도 가능한 신속하게 클라우드 컴퓨팅 기반 에뮬레이션 전략을 활용한 전자기록 장기보존 방안을 연구하는 것이 필요하다.



〈그림 2〉 bwFLA 시스템 구조와 서비스 제공

*출처: bwFLA, http://eaas.uni-freiburg.de/

2) VNC는 Virtual Network Computing의 약자로 RFB(Remote Framebuffer Protocol)를 이용하여 원격으로 다른 컴퓨터를 제어하는 시스템임.

3) WS-API는 Web Service Application Programming Interface의 약자로 Lua 웹 응용 프로그램에서 웹서버를 추상화한 API임.

http://ras.jams.or.kr http://dx.doi.org/10.14404/JKSARM.2019.19.4.001

2. 이론적 배경

2.1 장기보존 전략

마이그레이션(Migration), 에뮬레이션(Emulation), 인캡슐레이션(Encapsulation)은 대표적인 전자기록의 3대 장기보존 전략이다(DPT, 2006; 임진희, 2013). 먼저 마이그레이션은 기술의 발전, 시스템 노후화로 인하여 운영체제 및 소프트웨어 업그레이드, 시스템 교체 및 업그레이드 등을 수행할 때 전자기록물이 온전하게 접근 및 재현이 가능하도록 기존의 디지털 객체를 안정적인 형식으로 변환하는 전략을 뜻한다. 또한, 현재의 파일 포맷 방식이 새로운 시스템에서 재현이 불가능해 지거나 새로운 버전으로 업그레이드 될 경우에 오랜 시간이 지난 이후에도 재현 확률이 가장 큰 안정적인 파일 포맷으로 변환하는 방식이다. 예를 들어, 온-나라 문서 2.0에서 아래아 한글 또는 MS Word로 생성되었던 전자문서가 영구기록물관리시스템으로 이관될 때, 장기보존 되기 전에 문서보존 포맷인 pdf/a 파일로 변환하는 것은 우리나라의 장기보존 전략이 마이그레이션이기 때문이다. 시스템 전체를 유지해야 하는 에뮬레이션 전략보다 경제적인 비용과 기술적인 면에서 장점이 있다. 그러나 안정적인 포맷을 확보하기 위해 전자기록 유형별 그리고 파일 포맷별로 시스템 변경이나 소프트웨어 업그레이드에 대한 지속적인 모니터링이 필요하다. 또한, 마이그레이션은 근본적으로 포맷 변환이 이루어질 때 비트스트림 손실이 발생한다. 이로 인해 일부 본래의 모습이나 기능이 재현이 불가능할 수 있어 원본의 룩앤필(Look and Feel)이 달라질 수 있으며 결국, 마이그레이션은 진본성을 손상시킬 가능성이 있다는 본질적인 한계를 가지고 있다.

에뮬레이션은 하나의 컴퓨터 안에 전자기록물이 최초로 생산되었을 때와 동일한 형태의 하드웨어 및 소프트웨어로 구성된 가상의 다른 컴퓨터를 실행시키고 그 가상의 컴퓨터에서 전자기록물을 재현할 수 있도록 하는 보존전략이다. 〈그림 3〉은 Windows 10에서 VMWare Workstation Pro를 실행시켜 (1) 리눅스(Ubuntu 16.04 64bit), (2) Windows XP, (3) Windows 7의 세 개의 가상 컴퓨터를 실행시킨 화면을 보여 준다.

에뮬레이션은 가상의 컴퓨터를 실행시킬 수 있는 에뮬레이터(Emulator)란 소프트웨어가 필요하다(김명훈 외, 2013). VMWare, VirtualBox, Hyper-V 등이 대표적인 상용 에뮬레이터이다. 에뮬레이션 전략은 파일포맷의 버전 업그레이드가 필요 없고 원본 그대로의 모습을 재현할 수 있다는 큰 장점이 있다. 반면, 에뮬레이터와 함께 다양한 운영체제와 응용 프로그램을 담고 있는 대용량 시스템(템플릿) 이미지 파일을 보관하고 유지하기 위한 비용과 가상화 기술을 보유하고 있는 전문 인력 등이 요구된다. 그 외에도 시스템 노후화로 인하여 파일, 에뮬레이터, 시스템 이미지 파일의 저장매체 이관 작업이 여전히 필요하다.

대부분의 아카이브 기관에서는 마이그레이션과 에뮬레이션 중에서 하나를 주전략으로 채택하거나 하나는 주전략, 다른 하나는 부전략으로 사용하고 있다. 소정의, 한희정, 양동민(2018)에서 조사한 5개의 국립 아카이브 기관 중에서 4개 기관이 마이그레이션을 1개 기관에서는 에뮬레이션을 주전략으로 채택하고 있었다.

http://ras.jams.or.kr http://dx.doi.org/10.14404/JKSARM.2019.19.4.001

〈그림 3〉 VMWare Workstation Pro 실행 화면

마지막으로 인캡슐레이션 전략이다. 인캡슐레이션은 관련 메타데이터와 무결성 메시지를 원본 문서와 함께 하나의 개체로 패키징(Packaging)하는 과정이다. 이는 향후 메타데이터를 통해 기록물을 이해하고 신뢰성과 진본성을 제공하는데 도움을 줄 수 있으며, 무결성 메시지를 통하여 기록의 무결성과 진본성을 검증할 수 있다. 인캡슐레이션 전략은 마이그레이션 또는 에뮬레이션 전략을 선택하는 것과 상관없이 적용할 수 있으며 전자기록물의 신뢰성, 진본성을 제공하기 위한 가장 일반적인 방법으로 대부분 아카이브 기관에서 도입하고 있다.

현재 국가기록원 장기보존 주전략은 마이그레이션이다. 마이그레이션은 기능보다는 원본의 내용, 문맥, 외관 등에 대한 특성들을 보존해야 하는 전자문서 유형 장기보존에는 상당히 최적화 되어 있다. 그러나 기록물의 기능을 재현하거나 멀티미디어, 하이퍼텍스트 등의 장기보존에는 마이그레이션 전략이 적합하지 않다. 그러므로 전자문서 이외 전자기록에 대해 에뮬레이션 전략으로 장기보존하는 방안을 모색하여 마이그레이션과 함께 에뮬레이션을 장기보존 주전략의 하나로 추가하는 것이 필요하다.

2.2 클라우드 컴퓨팅(Cloud Computing)

클라우드 컴퓨팅은 4차 산업혁명 시대의 기반 인프라 기술로 인식되고 있다. 클라우드 컴퓨팅은 언제 어디서나 필요한 만큼의 컴퓨팅 자원을 필요한 시간만큼 인터넷을 통하여 활용할 수 있는 컴퓨팅 방식이다. 예를 들어, 빅데이터 및 인공지능 분야에 뛰어든 스타트업 기업이 초창기부터 빅데이터의 수집, 저장, 분석을 위한 방대한 컴퓨팅 자원과 인공지능 개발을 위한 슈퍼컴퓨터 등을 자체적으로 구매하고 유지관리 하는 것은 비현실적이다. 이때, 스타트업 기업이 실제 서버나 네트워크 장비를 구매하지 않고, 클라우드 컴퓨팅 서비스를 통해서 필요한 만큼만 비용을 지불하고 컴퓨팅 자원을 사용한다면 엄청난 경제적인 효과를 얻을 수 있을 것이다. 이처럼 자본력이 부족한 중소기업이나 스타트업은 클라우드 컴퓨팅 서비스를

활용하면 대규모 컴퓨팅 자원을 저렴하게 활용할 수 있다. 가트너(Gartner)에서는 2021년까지 공용 클라우드 시장이 매년 17.6%씩 성장할 것으로 예상하고 있으며, IaaS(Infra structure As A Service)와 SaaS(Software As A Service) 분야가 가장 빠르게 성장할 것으로 전망되었다(강맹수, 2019).

클라우드 컴퓨팅 개념은 1970년부터 조금 다른 의미로 또는 일부 개념으로 언급되었고, 그 실체가 드러난 것은 2000년 이후이다. 특히, 아마존의 AWS(Amazon Web Services)와 마이크로소프트의 애저(Azure)가 대중적으로 사용되면서 클라우드 컴퓨팅 분야는 급격히 발전했다. 대부분 기업과 개인의 민간 분야에서 활발하게 활용되었으며, 최근에는 공공 분야까지 도입되었다. 국가정보자원관리원에서 운영하고 있는 G-클라우드(구, 정부통합전산센터)가 우리나라의 대표 공공 클라우드 컴퓨팅 서비스이며, 중앙부처 및 산하 공공기관까지 각 기관별로 운영하는 시스템들을 G-클라우드로 이관하고 있다. 행정기관의 클라우드 전환율은 2016년 25.80%, 2017년 32.93%, 2018년 39.96%으로 점점 늘어가고 있는 추세이다. 또한 클라우드의 주요 특징은 〈표 2〉와 같이 정리할 수 있다.

클라우드 컴퓨팅 기술은 가상화(Virtualization) 기술과 분산처리(Distributed Processing) 기능이 핵심이다. 가상화는 컴퓨터에서 컴퓨터 자원의 추상화(Abstraction)하는 것을 말하는 상당히 포괄적인 용어이다. 가상이라는 단어에서 알 수 있듯이 실제로 존재하지 않으나 존재하는 것처럼 만들어서 실행하여 보여준다는 의미이다. 클라우드 컴퓨팅에서의 가상화는 CPU, 메모리, 디스크, OS 등을 구매하지 않아 실제로는 즉, 물리적으로 내 손안에 존재하지 않지만 마치 그러한 컴퓨팅 자원이 존재하는 것처럼 가상적으로 소프트웨어를 이용하여 만들어 주는 기술이다. 마법처럼 완전히 새로운 컴퓨팅 자원을 창조해 내는 것이 아니라 이미 존재하고 있는 컴퓨터의 자원을 일부 빌려오는 형식으로 구현한다. 클라우드 컴퓨팅에서의 분산처리 기술은 여러 개의 컴퓨터를 하나로 만들어 주는 역할을 한다. 사용자 측면에서는 클라우드 컴퓨팅을 구성하는 다수 서버에 대한 정보 없이 외부에 드러나 있는 하나의 창구를 통해서

〈표 2〉 클라우드 컴퓨팅의 특징

구분	특징
접속 용이성	• 시간과 장소에 상관없이 인터넷을 통해 이용 가능 • 클라우드에 대한 표준화된 접속을 통해 다양한 기기로 이용 가능
유연성	• 갑작스러운 이용량 증가나 이용자 수 변화에 신속하고 유연하게 대응 가능
주문형 셀프서비스	• 이용자는 서비스 제공자와 직접적인 상호작용을 거치지 않고, 자율적으로 자신이 원하는 클라우드 서비스를 이용 가능
사용량 기반 과금제	• 이용자는 서비스 사용량에 대해서만 비용을 지불 • 전기, 수돗물처럼 개인의 사용량에 따라 과금하는 방식
가상화와 분산처리	• 하나의 서버에서 다양한 컴퓨팅 자원을 나누어서 쓰는 가상화 기술 • 여러 대의 서버를 하나로 묶어 운영하고, 사용자의 컴퓨팅 자원 요구를 여러 서버에 분산처리함으로써 시스템 과부하를 최소화하는 분산처리 기술

* 출처: (강맹수, 2019)

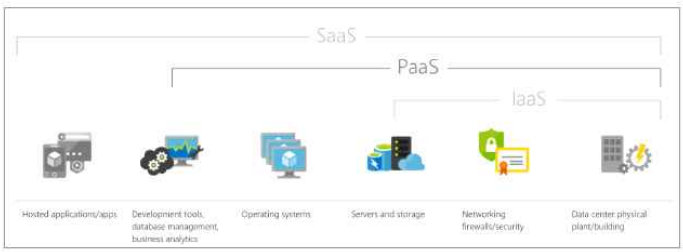
http://ras.jams.or.kr http://dx.doi.org/10.14404/JKSARM.2019.19.4.001

서비스를 제공받을 수 있으며, 관리자 측면에서도 각 서버를 별도 관리하지 않고 하나로 묶어서 감시하고 관할 수 있다. 이를 위해 분산처리 기술은 다수 서버의 컴퓨팅 자원(CPU, 메모리, 디스크, 네트워크 등)을 하나의 Pool로 구성하고, 사용자 요청이 오면 알맞게 요청이 분산되어 수행되도록 로드 밸런싱(Load Balancing)하여 제공한다.

클라우드 컴퓨팅 서비스는 유형에 따라 IaaS(Infrastructure As A Service), PaaS(Platform As A Service), SaaS(Software As A Service)로 구분된다(〈그림 4〉 참고).

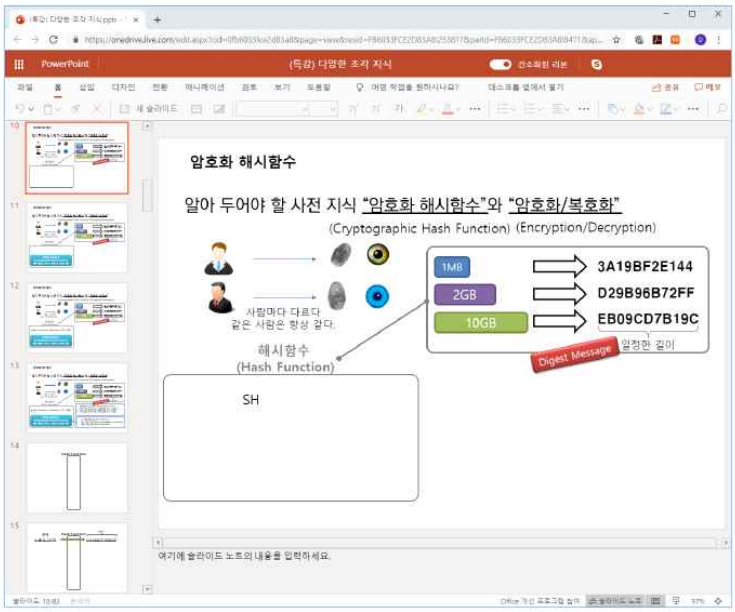
IaaS가 하드웨어라면, PaaS와 SaaS는 소프트웨어를 제공하는 서비스이다. 먼저, IaaS는 CPU, 메모리, 디스크 등 실제로 물리적인 하드웨어를 '구매하지 않고도' 마치 하드웨어를 가지고 있는 것처럼 서비스를 제공한다. 실제 하드웨어는 내 손안에 있지 않고 클라우드에 있으며, 원격지에서 인터넷을 통해 제어하여 동작하게 할 수 있다. 이를 통해서, 일반적으로 서버나 저장소, 네트워크를 필요에 따라 이용할 수 있게 서비스를 제공받을 수 있으며 서버와 저장소, 백업 인프라 구축 시 이용할 수 있다. PaaS는 운영체제, 개발 환경을 위한 시스템 소프트웨어, 데이터베이스와 같은 미들웨어 등에 해당하는 플랫폼 수준의 소프트웨어를 제공하는 서비스이다. 이러한 플랫폼을 실제로 본인의 서버나 PC에 '설치하지 않고' 클라우드 컴퓨팅을 통해 설치된 플랫폼 서비스를 제공받을 수 있다. 사용자는 인터넷으로 클라우드 컴퓨팅 서비스에 접속하여 플랫폼을 원격으로 제어하여 실행하고 결과를 볼 수 있다. SaaS는 네트워크를 통해 원하는 소프트웨어를 본인의 서버나 PC에 '설치하지 않고' 클라우드 컴퓨팅을 통해 서비스를 받을 수 있다. Google과 MS에서는 각각 Google Document(Spreadsheet, Presentation 등)와 MS Office(Powerpoint, Excel 등)을 설치 없이 인터넷 브라우저에서 직접 작성하여 저장할 수 있는 서비스를 제공하고 있다(〈그림 5〉 참고).

클라우드 컴퓨팅은 사용자가 실제 실물의 컴퓨터가 없는데도 불구하고 여러 개의 컴퓨팅 환경을 사용할 수 있게 해준다. 이는 다양한 하드웨어를 추상화하는 가상화 기술과 가상화된 하드웨어 위해서 운영체제를 실행해 주는 에뮬레이터 기술 덕분이다. 이러한 장점으로 기록



〈그림 4〉 제공 서비스 유형에 따른 클라우드 컴퓨팅 서비스

*출처: Microsoft Azure 홈페이지



〈그림 5〉 Chrome에서 실행시킨 MS Office 365 - 파워포인트

관리 분야뿐 아니라 대부분의 정보화 시스템을 운영하는 기관에서는 클라우드 컴퓨팅 환경으로 시스템들 이전하고 있다. 그러나 현재 가상화 기술이 지금까지 나와 있는 모든 하드웨어를 추상화하지 못한다. 가상화 기술이 지원할 수 있는 CPU 아키텍쳐 구조들이 제한적이기 때문이다. 예를 들어, 본 연구에서 테스트베드를 구축할 때 사용되는 가상화 기술인 KVM은 Intel 및 AMD에서 개발한 x86 CPU 프로세서를 지원하지만 Sun SPARC, DEC Alpha, IBM S/390 등의 마이크로프로세서를 지원하지 않는다. 그러므로 클라우드 컴퓨팅 환경의 장점을 전자기록의 에뮬레이션 장기보존 방안에 최대한 활용하면서도 한계점을 파악하여 에뮬레이션 전략으로 해결할 수 없는 부분을 정확히 도출하여야 한다.

2.3 가상화(Virtualization)

가상화 기술은 실제로 존재하지 않으나 존재하는 것처럼 가상으로 만들어 주는 기술이다. 컴퓨터 분야에서는 실제로 컴퓨터 자원(CPU, 메모리, 디스크 등)을 실제로 구매하지 않아도 자신의 컴퓨터 상에서 실제로 존재하는 것처럼 만들어 준다. 이들은 소프트웨어 기술로 구현되는데, 각자의 컴퓨터에 가상화를 지원하는 소프트웨어(VMware, Virtualbox 등)를 설치하고 Window OS 설치 파일 또는 리눅스 OS

설치 파일을 가상화 소프트웨어와 연결하면 〈그림 3〉처럼 가상화 소프트웨어에 설치되어 컴퓨터 안에 또 하나의 컴퓨터를 가지게 된다. 여기서 물리적으로 존재하는 컴퓨터(OS)를 호스트 컴퓨터(호스트 OS)라고 하고, 호스트 컴퓨터에 가상화 소프트웨어 도움을 받아 설치된 컴퓨터(OS)를 게스트 컴퓨터(게스트 OS)라고 한다. 예를 들어, 구 버전인 Windows XP만 실행되는 응용 프로그램을 실행시키고 싶은데 현재 본인이 가지고 있는 컴퓨터(호스트 컴퓨터)의 OS는 Windows 10인 경우에 가상화를 지원하는 소프트웨어를 설치하고 Windows XP 설치 파일을 해당 가상화 소프트웨어와 연결하여 설치한 후 가상화 소프트웨어의 도움을 받아 Windows XP를 시작하고 실행을 원하는 응용 프로그램을 실행시킬 수 있다. 새로운 컴퓨터를 갖게 되는 모습을 보이지만 가상화 소프트웨어가 실행되는 호스트 컴퓨터의 컴퓨팅 자원을 나눠 쓰는 것이기 때문에 부모 컴퓨터 이상의 컴퓨팅 자원을 사용할 수 없다.

이처럼 가상화 기술은 호스트 컴퓨터의 컴퓨팅 자원을 사용하는데, 호스트 컴퓨터의 컴퓨팅 자원을 사용하기 위해서는 호스트 컴퓨터의 컴퓨팅 자원을 논리화 및 추상화를 해야 한다. 그래서 가상화를 추상화(Abstraction)로 일컫기도 한다. 먼저, 논리화는 하나의 컴퓨팅 자원 작은 논리 단위로 쪼개져 있게 만드는 것이고, 추상화는 컴퓨팅 자원과 연관된 복잡하고 물리적인 다른 요소들을 안 보이게 하여 단순화시키는 기술을 얘기한다. 예를 들어, 호스트컴퓨터에 삼성전자에서 구매한 256GB(DDR4, 2,666MHz) 메모리 2개가 장착되어 있다고 가정하자. 이때, 논리화는 이 메모리를 1GB짜리 512개(2*256개)가 있는 것으로 만들어 주는 기술이고, 추상화는 이 메모리의 제조업체 무엇이고 어떤 기술로 제조됐는지 상관없이 '512GB 메모리(1GB 단위)'로 보여 줄 수 있게 하는 기술이다.

가상화 기술의 논리화 및 추상화를 수행하는 핵심 기능을 하이퍼바이저(Hypervisor)라고 한다. 논리화 및 추상화를 수행하고, 이를 통해 호스트 컴퓨터에서 다수의 게스트 OS를 동시에 실행할 수 있다. 하이퍼바이저는 〈표 3〉처럼 Type 1과 Type 2로 구분된다.

가상화 기술은 클라우드 컴퓨팅 분야와는 다른 연구 진영에서 연구되어 온 것으로 하나의 호스트 컴퓨터에서 여러 개의 게스트 OS를 실행하기 위해 개발되었다. 여러 개의 호스트 컴퓨터를 하나로 묶고, 여러 개의 게스트 OS를 실행하는 클라우드 컴퓨팅 서비스는 가상화 기술이 필요하고 호스트 컴퓨터와 OS에 최적화

〈표 3〉 하이퍼바이저 종류

구분	특징	종류
Type I	• OS가 프로그램을 제어하듯이 하이퍼바이저가 해당 하드웨어에서 직접 실행되며 게스트 OS는 하드웨어 위에서 2번째 수준으로 실행됨	MS Hyper-V VMWare ESX/ESXi Readhat KVM
Type II	• 하이퍼바이저는 일반 프로그램과 같이 호스트 운영 체제에서 실행되며 하이퍼바이저 내부에서 동작되는 게스트 OS는 하드웨어에서 3번째 수준으로 실행	MS Virtual PC VMWare WorkStation

http://ras.jams.or.kr http://dx.doi.org/10.14404/JKSARM.2019.19.4.001

된 가상화 기술을 활용하는 구조이다.

이중 KVM(Kernel-based Virtual Machine)은 리눅스 진영에서 구축한 오픈소스 기반 가상화 기술이다. 일단 오픈소스이기 때문에 저작권 문제와 추후 재현하지 못하는 것에 대한 위험이 적다. 또한, Type 1 이기 때문에 직접적으로 하드웨어와 통신하기 때문에 Type 2 보다 실행이 빠른 편이다. 이러한 이유로 본 연구에서는 클라우드 컴퓨팅 환경을 구축하기 위한 가상화 기술로 KVM를 채택한다. 그리고 KVM과 가장 최적화되어 있는 QEMU를 채택하였다. 단, 2007년 이후에 출시된 리눅스 버전에서 하드웨어 가상 머신(HVM: Hardware Virtual Machine) 기능을 지원하는 x86 CPU 프로세서에 설치하여야 한다는 조건이 있다. 그래서 유닉스 계열의 OS가 주로 운영되었던 Sun SPARC, DEC Alpha, IBM S/390 등의 CPU 프로세서를 지원하지 않는다. 예전에 Sun Microsystems에서는 Solaris라는 UNIX 계열 OS를 개발했는데, 이는 x86 CPU 프로세서까지 지원하기 때문에 KVM에서 가상화할 수 있는 OS이다.

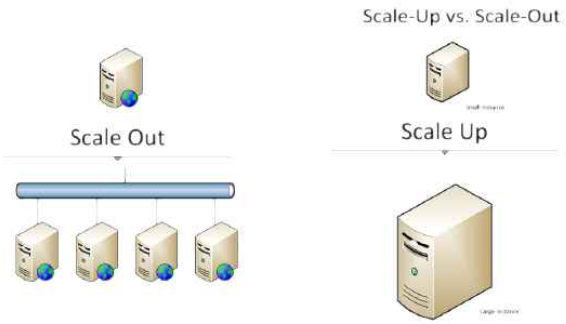
2.4 오픈스택(OpenStack)

클라우드 컴퓨팅 기술은 가상화와 분산처리가 핵심기능이다. 또한, 클라우드 컴퓨팅 서비스를 제공하기 위해서는 이 2개 핵심기능 이외에 다양한 기능들(사용자 인터페이스, 서비스 프로비저닝, 자원 유틸리티, 보안 및 프라이버시 등)을 포함하여야 한다. 클라우드 컴퓨팅 전문 기관과 기업들은 오픈소스 프로젝트 팀을 구성하여 가상화 기능은 별도로 개발하지 않고 앞서 2.3에서 소개되었듯이 연구되어 온 가상화 기술들 중에서 가장 적합한 기술을 선택할 수 있는 유연한 구조로 설계하고, 분산처리를 포함한 다양한 기능들은 자신만의 방법으로 개발하여 클라우드 컴퓨팅 서비스를 제공하기 위한 플랫폼을 구축하여 왔다.

클라우드 컴퓨팅에서의 분산처리 기술은 여러 개의 컴퓨터를 하나로 만들어 준다. 기존 단일 서버 중심의 컴퓨팅 구조에서는 사용자의 수가 증가하여 서버가 서비스를 제공해야 하는 규모가 커지면 서버의 용량을 늘리기 위해서 기존 서버를 폐기하고 용량이 큰 새로운 서버를 도입하는 Scale-up 방식이었다. 그러나 클라우드 컴퓨팅에서는 분산처리 기능 덕분에 기존의 서버 Pool에 새로운 서버를 추가하는 Scale-out 방식으로 동작한다(〈그림 6〉 참고).

컴퓨팅, 저장장치, 네트워크 자원 등으로 구성된 대규모 자원 Pool을 제어하고 관리하고, 웹 기반의 사용자 인터페이스를 통해 관리자는 손쉽게 자원을 제어하고 서비스를 관리할 수 있으며, 서비스 이용자는 클라우드 컴퓨팅 서비스를 동적으로 제공받고 관리할 수 있다. 이러한 소프트웨어의 역할이 마치 하나의 컴퓨터에서 자원을 할당하고 스케줄링하는 운영체제와 유사하게 여러 개의 서버에 대해서 하기 때문에 '클라우드 운영체제'라고 불리운다.

대표적인 소프트웨어로는 Cloud.com에서 개발한 CloudStack, UC Santa BarBara에서 개발한 Eucalyptus, 현재는 OpenNebula Systems에서 유지관리하고 있는 OpenNebula, NASA와 Rackspace가 2010년 7월에 시작한 OpenStack(현재는 OpenStack Foundation에서 운영)이 있다. CloudStack을 제외하고 대부분 오픈소스 프로젝트로 운영되고 있으며, 그 중에서도



〈그림 6〉 Scale-up과 Scale-out 개념
* 출처: IDCHOWTO, https://idchowto.com

오픈스택은 다른 공개 소프트웨어처럼 기업이 비교적 자유롭게 쓸 수 있다. OpenStack에는 컴퓨터, 저장장치, 네트워크 장비 제조사와 소프트웨어 솔루션 업체 그리고 정보통신 서비스 사업자를 아우르는 수많은 정보통신 글로벌 선도 기업들이 회원사로 등록되어 있어 모든 클라우드 컴퓨팅 오픈소스 프로젝트 중에서 가장 많은 기업이 프로젝트에 참여하고 있다. 이와 더불어 사용자뿐만 아니라 오픈소스에 기여하는 개발자 층도 전 세계적으로 지속적으로 확대되고 있어, 오류 발생 시 빠르고 능동적인 대처가 이루어지고 있다.

2010년 Austin 이라는 이름의 배포판을 시작으로 알파벳 순서에 따라 공식 발표되던 OpenStack은 Diablo 배포판 이후 6개월마다 배포판을 발표하고 있다. OpenStack의 배포과정은 계획, 개발, 사전배포, 공식배포의 4단계로 이루어진다. 매번 새로운 배포판이 공식 발표된 이후에는 차기 배포판에 대한 개발 논의가 Design Summit을 포함해 약 4주간 지속되며, 이 기간 동안 수집된 각종 기능 추가 또는 성능 개선 현안들에 대한 검토를 거쳐 우선순위를 부여하고 전체적인 개발일정 계획을 수립한다. 이후에는 수립된 일정 계획에 맞춰서 선별된 기능 추가 또는 성능 개선 현안에 대한 개발이 동시다발적으로 진행된다(박종근, 최강일, 이상민, 이정희, 이범철, 2013).

OpenStack은 주요 기능이 몇 개의 세부 프로젝트로 나뉘어 개발되고 있다. 초기에는 컴퓨팅 서비스인 Nova, 오브젝트 저장장치 서비스인 Swift, 템플릿 이미지 관리 서비스인 Glance의 3개의 프로젝트로 구성되었다. 계속적으로 버전이 업데이트되고 기능이 확장되면서 새로운 프로젝트가 지속적으로 추가 발표되어 점차 그 영역을 확대되고 있다. 예를 들어, 네트워킹 서비스와 볼륨 저장장치 서비스와 같이 Nova에서 함께 제공되던 서비스가 별도의 프로젝트로 분리되거나 인증 서비스 또는 사용자 인터페이스와 같이 독립된 프로젝트가 새롭게 추가되기도 한다.

OpenStack은 라이센스 비용없이 무료로 사용할 수 있는 오픈 소스이며, 많은 회사와 개인

개발자들이 참여하여 프로젝트에 기여하고 있기 때문에 지속적인 업데이트가 가능하다. 그렇기 때문에 OpenStack은 현재 클라우드 컴퓨팅 플랫폼 중에서는 가장 주목 받고 있다. 본 연구팀은 현재 대규모 시스템 시장에서도 꾸준히 시장 점유율을 높이고 있는 Linux 처럼 기본적으로 무료이고, 지속적인 발전으로 완성도가 높아질 가능성이 많기 때문에 OpenStack가 업계 표준이 될 것으로 예상하여, 이 본 연구에서는 클라우드 컴퓨팅 플랫폼으로 OpenStack을 활용한다.

2.5 P2V(Physical-to-Virtual) 변환기

P2V(Physical-to-Virtual) 변환기는 실제로 존재하는 물리(Physical) 서버에서 클라우드 컴퓨팅 환경의 가상(Virtual) 서버로 시스템을 원래의 모습 그대로 이전하는 도구를 의미한다. 〈그림 7〉은 P2V 변환기를 활용하여 물리서버(〈그림 7〉에서 Source Server)를 가상서버(〈그림 7〉에서 Target Server)로 이전하는 과정을 보여 준다. P2V 변환기는 기존의 레거시(Legacy) 시스템을 클라우드 컴퓨팅 환경으로 시스템 전체를 이전할 때 주로 사용되고 있다. 먼저 물리서버에 P2V 변환기 소프트웨어를 설치하고, 클라우드 컴퓨팅 환경에 동일한 운영체제를 가진 가상서버를 생성하고, 네트워크를 통해 물리서버에서 가상서버로 시스템 데이터를 이전하는 과정을 거친다.

〈표 4〉는 현재 많이 사용되고 있는 P2V 변환기 제품들을 비교하였다. 크게 무료와 유료로 구분된다. 무료 제품은 경제적인 비용이 거의 없다는 장점이 있지만 지원 OS, 지원 클라우드/하이퍼바이저 플랫폼이 제한적이며, 오류 상황이 발생하더라도 해결책을 쉽게 얻을 수 없다. 유료 제품은 경제적인 비용은 들지만 시스템 이전이 발생하는 오류에 대해 기술지원을 받을 수 있기 때문에 안정적으로 시스템 이전을 수행할 수 있다는 장점이 있다. 그래서 본 연구에서는 무료 제품 중에서는 virt-p2v를, 유료 제품 중에서는 국내 기업의 zConverter를 활용하여 실험을 진행한다.

〈그림 7〉 P2V 변환기 시스템 이전 과정

* 출처: PlanetVM, https://planetvm.net/

http://ras.jams.or.kr http://dx.doi.org/10.14404/JKSARM.2019.19.4.001

〈표 4〉 주요 P2V 변환기 제품 비교

제품명	비용	지원 OS 계열	지원 클라우드/하이퍼바이저 플랫폼
vCenter Converter (VMWare 사)	무료	윈도우/리눅스	• VMware
Disk2vhd (Microsoft 사)	무료	윈도우	• VHD(MS Virtual PC or Hyper-V)
PlateSpin Migrate (Micro Focus 사)	유료	윈도우/리눅스	• VMware vCenter, ESXi, Hyper-V, KVM, Azure, VMware vCloud Director
V2V Converter (StarWind 사)	무료	윈도우	• KVM, VMware, Hyper-V
virt-p2v (Redhat 사)	무료	리눅스	• KVM, Openstack, oVirt, RHV
zConverter (ISA Technologies 사)	유료	윈도우/리눅스	• KVM, Xen, VMware, Hyper-V, Openstack, Azure, AWS, Cloudstack

3. 클라우드 컴퓨팅 기반 에뮬레이션 전략을 활용한 장기보존 방안

본 장에서는 실제 에뮬레이션 테스트베드를 구축하고 이를 기반으로 실험한 결과를 통해 클라우드 컴퓨팅 기반 에뮬레이션 전략을 활용한 전자기록 장기보존 방안을 제시하고자 한다.

3.1 에뮬레이션 테스트베드 구축

에뮬레이션 테스트베드 구축을 위해서 클라우드 컴퓨팅 환경을 〈그림 8〉과 같이 구성하였다. 테스트베드에는 2개의 서버(〈표 5〉 참고)와 2개의 네트워크 스위치로 이루어져 있다. 기본적으로 클라우드 컴퓨팅을 위해서는 최소 2개의 서버(관리 서버, 컴퓨팅 서버)와 2개의 스위치(관리 스위치, 통합 스위치)가 필요하다. 각 서버의 운영 체제는 CenOS 7(리눅스)를 설치하고, 하이퍼바이저는 KVM를, KVM 상에서 게스트 OS를 실행하는 역할을 담당하는 에뮬레이터는 QEMU를 설치하였다. 그리고 서버들을 하나의 Pool로 구성하기 위해 KVM과 QEMU와 연동하여 이들을 관리하고 자원을 할당하는 클라우드 운영체제는 OpenStack Queens 버전을 사용하였다. 이들을 관리자 또는 사용자가 관리하는 인터페이스는 〈그림 9〉에서 처럼 ㈜이노그리드에서 개발한 오픈스택잇(OpenStackit) 솔루션을 적용하였다.

에뮬레이션 테스트를 활용하여 네 가지 실험을 진행하였다. 첫 번째는 클라우드 컴퓨팅에 Windows XP를 설치하고 한컴오피스 97을 설치하고 한컴오피스 97버전의 파일(office97.hwp)을 확인하는 실험이다. 두 번째는 클라우드 컴퓨팅에 MS DOS를 설치하고 보석글 VersionG 1.01을 설치하고 보석글 파일(jewelry.twp)을 확인하는 실험이다. 첫 번째와 두 번째 실험은 전자기록을 에뮬레이션 전략으로 장기보존 주 전략 채택가능성에 대해 검토하기 위한 실험이다. 세 번째 실험은 특정 PC를 백업 도구를 사용하여 템플릿 이미지 파일을 만든 다음, 그 템플릿 이미지 파일을 클라우드 컴퓨팅으로 이관하여 설치하고, 특정 PC의 모습이 클라우드 컴

〈그림 8〉 에뮬레이션 테스트베드 클라우드 컴퓨팅 환경 구성

〈표 5〉 클라우드 컴퓨팅 구성 서버 사양

구분	CPU	메모리	볼륨 스토리지	이미지 스토리지	가상서버 스토리지
서버1(관리, 컴퓨팅)	24 Core	256GB	1TB	1TB	3TB(공유 스토리지)
서버2(컴퓨팅)	24 Core	256GB	-	-	
계	48 Core	512GB	1TB	1TB	3TB

〈그림 9〉 오픈스택잇의 메인 화면 및 컴퓨팅 서버 조회 화면

퓨터에 재현되는지를 검토한다. 네 번째 실험은 2개의 서버로 이루어진 시스템을 백업 도구를 사용하여 템플릿 이미지 파일을 만든 다음, 그 템플릿 이미지 파일을 클라우드 컴퓨팅으로 이관하여 설치하고, 2개의 서버로 이루어진 시스템이 클라우드 컴퓨터에 제대로 재현되는지를 검토한다. 세 번째와 네 번째 실험은 마이그레이션으로는 장기보존하기 힘든 전자기록의 기능을 전자기록을 에뮬레이션 전략 방식을 통해서 재현할 수 있는지 여부를 검토하는 실험이다. 〈표 6〉은 네 가지 실험에 대한 개요를 보여 준다.

〈표 6〉 클라우드 컴퓨팅 기반 에뮬레이션 실험 개요

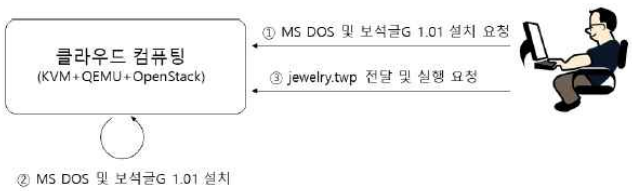
구분	사양		내용	실험방법	
MS DOS 및 보석글 에뮬레이션 실험 (3.2)	OS		MS DOS	클라우드 컴퓨팅 환경에 OS, SW설치 및 실행	
	SW		보석글G 1.01		
	전자기록		jewelry.twp		
XP 및 한컴오피스 에뮬레이션 실험 (3.3)	OS		Windows XP	클라우드 컴퓨팅 환경에 OS, SW설치 및 실행	
	SW		한컴오피스 97		
	전자기록		office97.hwp		
단일 서버 시스템 에뮬레이션 실험 (3.4)	OS		Ubuntu 14.04	zConverter P2V변환기	물리서버에서 클라우드 컴퓨팅 환경의 가상 서버로 시스템 이전 및 실행
	SW		MySQL 5.5		
다수 서버 시스템 에뮬레이션 실험 (3.5)	1	OS	Windows Server 2012 Standard R2	zConverter P2V변환기	
		SW	WebtoBv4.1, JEUS v6.0		
	2	OS	CentOS 7.6	virt-p2v P2V 변환기	
		SW	Oracle 10g		

3.2 MS DOS 및 보석글 에뮬레이션 실험

〈그림 10〉과 같이 클라우드 컴퓨팅 환경에 MS DOS를 설치 요청 및 설치 완료하고 전자문서(jewelry.twp)를 실행하여 jewelry.twp 파일을 보석글G version 1.01을 통해 확인한다.

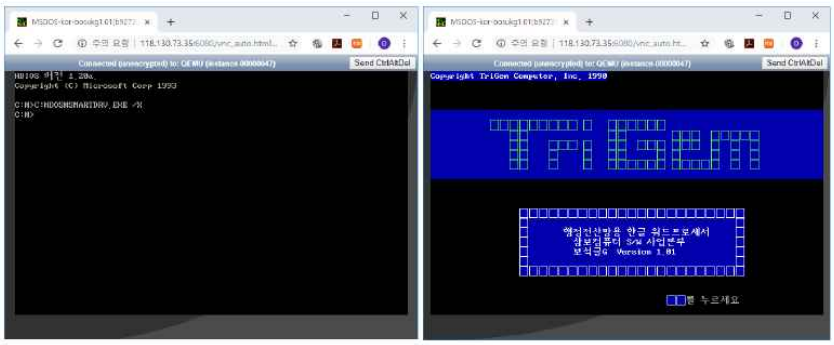
클라우드 컴퓨팅 환경에 MS DOS 설치 요청 및 설치 완료를 하고 전자문서(jewelry.twp)를 실행하여 jewelry.twp가 잘 실행이 되는지 확인한다. 〈그림 11〉은 MS DOS 설치를 요청하고 클라우드 환경에서 MS DOS가 설치된 후의 화면이다. 〈그림 12〉에서는 MS DOS 컴퓨팅 환경과(왼쪽)과 보석글 파일(오른쪽)을 각각 실행한 모습을 보여 준다.

MS DOS 및 보석글 에뮬레이션 실험을 통해서 구전자문서시스템에서 보석글로 생산된 공문서를 재현하여 확인할 수 있는 가능성을 볼 수 있다. 또한, MS DOS 이외에도 Windows 3.X도 에뮬레이션이 가능하다. 만약, 실시간으로 운영체제와 응용 소프트웨어 설치 요청이 처리가 가능하도록 서비스가 구축된다면 구전자문서시스템에 남아 있지만 확인이 힘든 다양한 전자문서들까지도 재현을 통해 확인할 수 있을 것으로 기대된다.



〈그림 10〉 전자기록 에뮬레이션 실험 과정

〈그림 11〉 클라우드 컴퓨팅 환경에 MS DOS가 설치된 모습



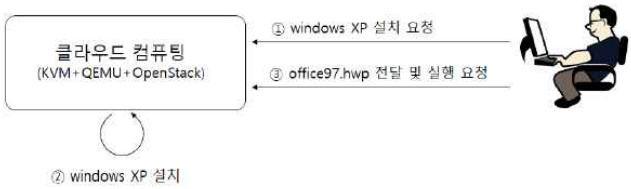
〈그림 12〉 클라우드 컴퓨팅 환경에 설치된 MS DOS와 보석글G 1.01 실행한 화면

3.3 XP 및 한컴오피스 에뮬레이션 실험

여기서는 〈그림 13〉과 같이 클라우드 컴퓨팅 환경에 Windows XP 설치 요청 및 설치 완료하고 전자문서(office97.hwp)를 실행하여 office97.hwp가 잘 실행이 되는지 확인한다.

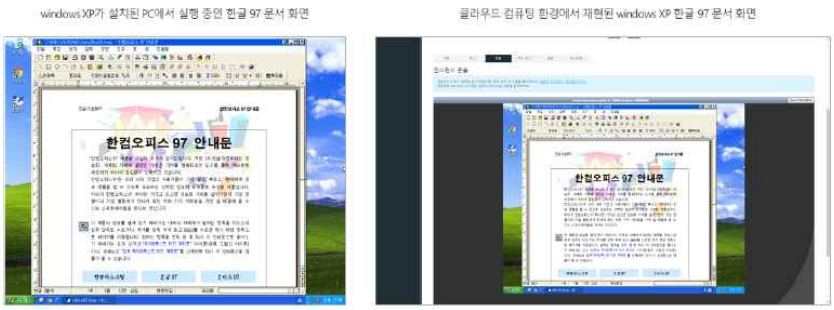
클라우드 컴퓨팅 환경에 Windows XP를 설치 요청 및 설치 완료를 하고 전자문서(office97.hwp)를 실행하여 office97.hwp가 잘 실행이 되는지 확인한다. 〈그림 14〉는 Windows XP 설치를 요청하고 클라우드 컴퓨팅 환경에서 Windows XP가 설치된 후의 화면이다. 〈그림 15〉에서는 사용자 컴퓨터와(왼쪽)과 클라우드 컴퓨팅 환경(오른쪽)에서 각각 잘 실행되고 있는 모습을 보여 준다.

http://ras.jams.or.kr http://dx.doi.org/10.14404/JKSARM.2019.19.4.001



〈그림 13〉 전자기록 에뮬레이션 실험 과정

〈그림 14〉 클라우드 컴퓨팅 환경에 Windows XP가 설치된 모습(OpenStackit 화면)



〈그림 15〉 원본PC와 클라우드 컴퓨팅 환경 가상서버에 설치된 Windows XP와 실행된 한글 문서(office97.hwp)

XP 및 한컴오피스 에뮬레이션 실험을 통해서도 3.2에서의 실험과 같이 오랫동안 확인이 힘들었던 다양한 전자문서들을 에뮬레이션 방식으로 확인할 수 있음을 기대할 수 있다.

3.4 단일 서버 시스템 에뮬레이션 실험

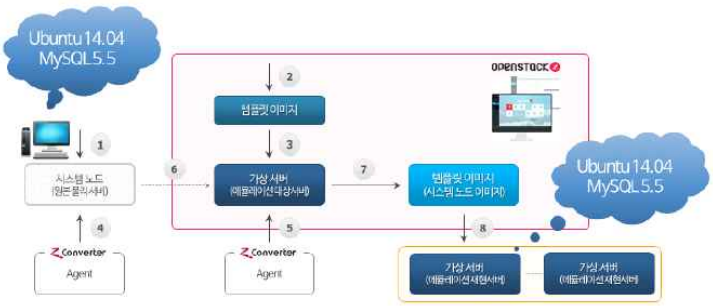
세 번째 실험은 사용자 컴퓨터에서 실제 운영하고 있는 서버를 클라우드 컴퓨팅 환경의

http://ras.jams.or.kr http://dx.doi.org/10.14404/JKSARM.2019.19.4.001

가상서버로 이관하는 실험이다. 실험은 〈그림 16〉과 같은 절차를 통해 이루어진다. ① 에뮬레이션 대상 원본물리서버(MySQL 5.5가 설치된 Ubuntu 14.04)를 준비한다. ② 원본물리서버와 같은 운영체제(Ubuntu 14.04)로 제작된 템플릿 이미지(설치파일)을 준비한다. ③ 제작된 템플릿 이미지로 클라우드 컴퓨팅 환경에 가상서버를 생성한다. ④ 원본물리서버에 zConverter Agent를 설치한다. ⑤ 가상서버에도 zConverter Agent를 설치한다. ⑥ zConverter로 원본물리서버에서 가상서버로 데이터를 이관한다. ⑦ 가상서버에 정상적으로 이관되었는지 확인하고 백업 및 스냅샷 기능으로 템플릿 이미지(설치파일)을 생성한다. 여기까지가 사용자 컴퓨터의 원본물리서버를 클라우드 컴퓨팅 환경으로 옮기는 과정이다. ⑧인터넷을 통해 사용자가 MySQL 5.5가 설치되어 있는 리눅스 확인 요청하면 템플릿 이미지(설치파일)에서 가상서버를 생성한다. 사용자는 원격으로 클라우드 컴퓨팅 환경에 있는 가상서버에 접속하여 MySQL 5.5가 실행되고 있는지 확인할 수 있다.

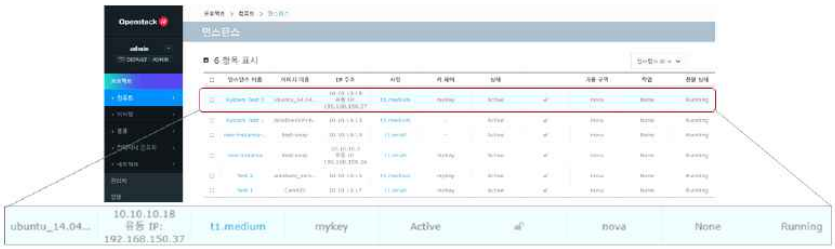
이 과정은 원본물리서버의 특정 파일이나 응용 프로그램을 옮기는 것이 아니라 시스템 자체를 옮기는 과정이다. 처음 컴퓨터에 OS를 설치할 때 CD 또는 USB에 파일로부터 설치한다. 즉, 시스템도 하나의 파일로 표현될 수 있다. 템플릿 이미지가 여기에 해당한다. 이러한 개념에 응용하여 시스템 전체를 다시 하나의 템플릿 이미지 즉, 하나의 파일로 만들 수 있으며 시스템 전체를 이관하는데 활용할 수 있는 것이다.

본 연구에서 템플릿 이미지 생성 이후의 과정은 ㈜이노그리드에서 개발한 OpenStackit에서 동작시키고 확인할 수 있다. 웹브라우저로 클라우드 컴퓨팅에 접속하면 〈그림 17〉의 OpenStackit 화면을 볼 수 있고, 프로젝트 → 컴퓨트 → 인스턴스 메뉴에서 클라우드 컴퓨팅 환경에는 원본물리서버와 동일한 가상서버가 설치되었음을 확인할 수 있으며, 클라우드 컴퓨팅 환경의 생성된 가상서버(리눅스 Ubuntu 14.04)을 부팅하여 MySQL 5.5에 접속한 모습과 이전 사용자 컴퓨터에서 부팅되어 MySQL 5.5가 동일하게 실행되고 있는 모습을 〈그림 18〉에서 확인할 수 있다. 가상서버 접근하는 방법은



〈그림 16〉 zConverter 기반 전자기록 시스템 에뮬레이션 실험 절차

http://ras.jams.or.kr http://dx.doi.org/10.14404/JKSARM.2019.19.4.001



〈그림 17〉 클라우드 컴퓨팅 환경에 템플릿 이미지 파일로 설치된 리눅스

사용자 컴퓨터(Linux)에서 실행중인 MySQL 접속 화면

클라우드 컴퓨팅 환경에서 재현된 Linux와 MySQL 접속 화면

〈그림 18〉 사용자 컴퓨터와 클라우드 컴퓨팅 환경에 설치된 원본물리서버와 가상서버

OpenStackit을 통해서 콘솔 화면을 통해서 가능하며, 리눅스 계열은 putty와 같은 SSH[4] 클라이언트로 접속하여 확인할 수도 있다.

이번 실험을 통해서 에뮬레이션 방식을 통해서 원본물리서버 시스템 전체를 본래의 특성(기능, 모습 등)을 모두 보존할 수 있는지의 가능성을 파악할 수 있었다. 앞서 3.2와 3.3의 실험에서는 전자기록(아래아 한글, 보석글 파일)를 확인할 수 있는 운영 환경에 대한 정보만을 이용하여, 클라우드 컴퓨팅 환경에 동일한 운영체제 및 구동SW를 설치한 다음 전자기록을 확인하였다. 반면, 이번 실험에서는 전자기록을 포함한 운영 환경 전체를 클라우드 컴퓨팅 환경에 이관하였으며 운영 환경 전체를 보존할 수 있다는 것을 확인하였다. 전자기록의 유형이 전자문서, 시청각기록물, 웹기록물, 행정정보데이터세트 등 다양한 형태로 확대되고 있으며, 디지털 객체는 물론 해당 객체를 구동하는 시스템 하드웨어 및 소프트웨어를 포함한 운영 환경까지 보존하는 것을 고려해야 하는 상황들

4) SSH는 Secure Shell의 약자로 네트워크를 통해 다른 컴퓨터에 로그인하거나 원격 시스템에서 명령을 실행하기 위한 응용 프로그램 또는 프로토콜임.

이 생기고 있는 상황에서 시스템 전체를 이관하는 방식은 전자기록물을 장기보존할 수 있는 좋은 해결책이 될 것으로 기대된다.

3.5 다수 서버 시스템 에뮬레이션 실험

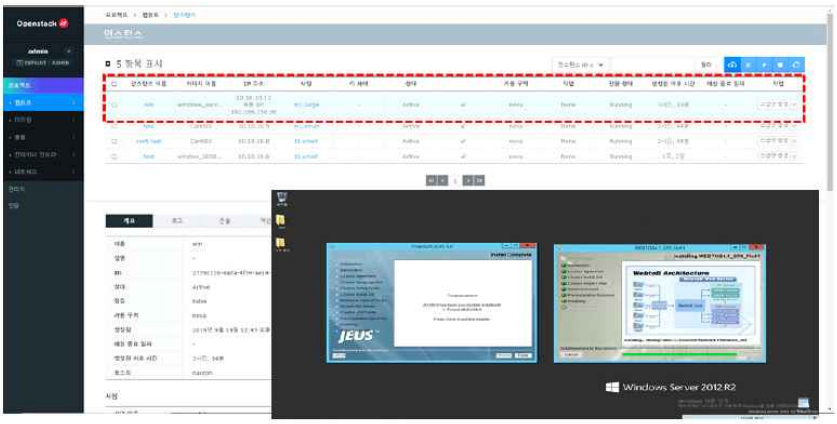
네 번째 실험은 2대의 서버가 서로 연결되어 기본적인 웹포털서비스를 제공하고 있는 시스템을 클라우드 컴퓨팅 환경의 가상서버로 이전하여 에뮬레이션을 수행하는 것이다. 시스템을 다른 환경으로 이관하면 하나의 서버에 대한 설정, 시스템 연계를 위한 설정, 외부 서비스와의 연계를 위한 설정 등이 필요하며, 설정 과정은 상황에 따라 다양하고 전문성이 필요하다. 이러한 상황을 실험하기 위해 〈그림 19〉와 같이 웹 포털서비스를 제공하는 물리서버 2대를 각각 zConverter와 virt-p2v를 사용하여 P2V 변환을 수행한 다음 클라우드 컴퓨팅 환경에 가상서버를 생성하는 과정을 통해 에뮬레이션을 수행한다.

먼저, 물리서버 1은 웹포털서비스의 입구에 해당하는 곳으로 Web서버와 WAS(Web Application Server)가 설치되어 있다. Web서버로는 WebtoB version 4.1이, WAS로는 JEUS version 6.0이 설치되어 있다. 물리서버 2는 웹포털 서비스의 자료를 보관하는 곳으로 데이터베이스(Oracle 10g)가 설치되어 있다. 물리서버 1은 3.4의 실험에서와 같이 〈그림 16〉의 에뮬레이션 절차를 통해 물리서버 1에서 가상서버 1로 시스템이 이전된다. 〈그림 20〉은 물리서버 1이 클라우드 컴퓨팅 환경에서 원래의 모습 그대로 가상서버 1에서 실행되고 있는 모습을 보여준다.

물리서버 2는 virt-p2v에 의해서 〈그림 21〉의 절차에 의해서 클라우드 컴퓨팅 환경으로 이전된다. ① 에뮬레이션 대상 물리서버 2(Oracle 10g가 설치된 CenOS 7.6)를 준비한다. ② 물리서버 2에 virt-p2v를 설치하고, 물리서버 2가 virt-p2v를 통해 부팅하여 물리서버 2 템플릿 이미지는 생성하여 중계서버에 템플릿 이미지를 전송한다. ③ 중계서버는 물리서버 2가 송신한 템플릿 이미지를 수신하고, 중계서버의 virt-p2v

〈그림 19〉 2대의 물리서버의 클라우드 컴퓨팅 기반 에뮬레이션 과정 개요

http://ras.jams.or.kr http://dx.doi.org/10.14404/JKSARM.2019.19.4.001



〈그림 20〉 클라우드 컴퓨팅 환경 가상서버 1에 설치된 물리서버 1 및 실행화면

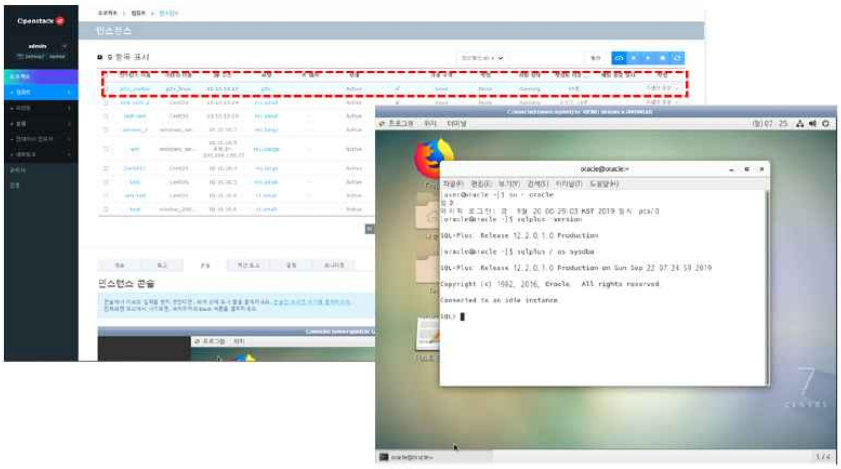
〈그림 21〉 virt-p2v 기반 전자기록 시스템 에뮬레이션 실험 절차

와 virt-managmer를 통해서 해당 이미지를 확인하고 클라우드 컴퓨팅 환경에 적합한 qcow2[5] 형태의 이미지를 생성한다. ④ 생성된 이미지는 virt-p2v에 의해서 클라우드 컴퓨팅 환경으로 이전된다. ⑤ 변환된 이미지로 클라우드 컴퓨팅 환경에서 가상서버 2를 생성한다.

〈그림 22〉는 물리서버 2가 클라우드 컴퓨팅 환경에서 원래의 모습 그대로 가상서버 2에서 실행되고 있는 모습을 보여준다.

지금까지 2대의 물리서버 1, 2를 클라우드 컴퓨팅 환경으로 이전하여 각각의 서버를 실행하는 데에는 성공하였지만 실제 웹포털서비스에는 접속되지 않았다. 물리서버 1과 물리서버 2가 설정되어 있는 네트워크는 192.168.100.* 였고, 물리서버 1의 IP주소는 192.168.100.101, 물리서버 2의 IP주소는 192.168.100.102로 설정되어 있다. 그리고, 2개의 물리서버가 클라우드 컴퓨팅 환경으로 이전되면서 가상서버 1에는 10.10.10.13, 가상서버 2에는 10.10.10.19의 IP주소를 설정하였다. 그래서 사용자가 가상서

5) QCOW2는 가상 머신 디스크 이미지 저장 형식입니다. QCOW는 QEMU Copy On Write의 약자임.



〈그림 22〉 물리서버 2가 클라우드 컴퓨팅 환경에서 에뮬레이션된 가상서버 2 및 실행화면

버 1의 Web서버를 접속하면 WAS를 통해서 가상서버 2의 DBMS에 접속하려고 할 때, 10.10.10.19로 접속하지 않고, 원본물리서버에 설정되어 있는 192.168.100.102로 접속을 시도한다. 그러므로 Web서버/WAS 응용 프로그램에서의 DBMS 접속 정보를 모두 192.168.100.102에서 10.10.10.19로 수정하였고 웹포털서비스는 정상 동작하였다.

이번 실험을 통해서는 전자문서를 에뮬레이션으로 확인하는 것을 넘어, 여러 서버로 구성된 시스템에서 생성되고 운영되는 행정정보데이터세트, 웹기록물까지도 원래의 모습과 기능 그대로를 에뮬레이션 방법으로 보존할 수 있다는 가능성을 확인할 수 있다. 본 실험에서 진행된 2개의 서버로 구성된 시스템은 DBMS와 Web서버/WAS와의 내부 서버들 사이의 연계를 위한 정보만 수정하면 되는 단순한 설정이었다. 그러나 시스템이 복잡해질수록 서버들에 대한 세부 설정, 특히, 외부 서비스와의 연계를 위한 설정은 더욱 복잡하고 어려운 작업이다. 향후에는 지속적으로 더욱 복잡한 구조의 시스템을 가정하여 테스트베드를 구축하여 실험함으로써 시스템 전체를 에뮬레이션하는 성숙된 기술을 확보해야 한다.

3.6 에뮬레이션 전략을 활용한 장기보존 방안

현재 우리나라는 전자문서, 행정정보데이터세트, 시청각기록물, 웹기록물 등 다양한 전자기록 유형 중에 전자문서 유형의 전자기록에 대해서는 마이그레이션 전략을 활용하여 체계적인 시스템을 구축하고 있다. 초기에는 전자문서의 비중이 높아졌으나 최근에는 행정정보데이터세트, 시청각기록물, 웹기록물의 비중이 높아지고 있지만 이에 대한 대응은 진행되고 있지만 많은 보완과 연구가 필요하다.

이런 상황에서 에뮬레이션 전략은 마이그레이션 전략을 대체 방안이 아니고 현재 마이그레이션 전략을 해결하지 못하거나 포용하지 못하는 전자기록을 장기보존하는 보완 방안으로 활용해야 한다. 그래서 본 연구에서는 다음과 같은 3개의 장기보존 활용방안을 제시한다.

첫 번째, 아카이브는 유형, 매체, 시대에 관계없이 생산된 다양한 기록을 수집하며 해당 기록의 진본성・신뢰성・무결성・이용가능성을 보장할 수 있어야 한다. 종이기록은 종이와 필기구의 수명을 고려하여 관리된다면 기록의 4대 속성이 보장될 수 있지만 다양한 매체에 수록된 전자기록은 다르다. 전자기록이 담긴 매체의 수명에 따라 잘 보존한다 하더라도 해당 매체를 읽을 수 있는 기계와 매체에 담긴 파일을 재생할 수 있는 애플리케이션이 없으면 읽고 해석할 수가 없다. 따라서 전자기록은 이용가능성 보장을 위해 별도의 조치가 필요한 것이다.

정부업무관리시스템을 사용하여 표준화된 전자문서를 사용하는 현재와는 달리 PC 도입 초창기에는 (구)전자문서시스템을 이용한 DOS 기반의 워드프로세스들이 공문서로 활용되었다. 에뮬레이션 대상으로 실험한 보석글은 당시 사용되었던 초기 전자문서 중 하나로 현재는 이를 재현할 애플리케이션의 공급이 중단된 상태이다. 또한 당시에 사용되던 OS와 저장매체를 읽을 수 있는 PC도 현재는 찾아보기 어려운 실정이다.

클라우드 컴퓨팅 기반 에뮬레이션 전략은 이처럼 과거에 사용하여 생산된 기록은 존재하지만 이를 재현하여 이용가능성을 보장할 수 있는 기술이 사라진 전자기록에 대해 필요한 전략이다. 1.3의 미국의 올리브와 독일의 bwFLA 역시 각각 CDROM에 저장된 애플의 Applie II, Macintoch, MS DOS, Windows 3.1 등의 예전 OS에서 예술작품, 예전 프로그램(게임, 브라우저, 워드프로세서 등)의 재현을 위해 추진된 클라우드 컴퓨팅 기반 에뮬레이션 프로젝트다.

두 번째, 행정정보데이터세트는 데이터베이스에 저장되어 있지만, 일반인들이 실제 데이터베이스에 접속하여 데이터를 직접 확인하지 않는다. 대부분 데이터베이스, 웹서버, WAS 등이 외부 응용 프로그램이나 솔루션과 연계하여 전자문서 또는 그래프 등의 형태로 확인할 수 있는 사용자 인터페이스를 제공한다. 예를 들어, 대학 학사정보시스템의 행정정보데이터세트는 데이터베이스 안에 있는 데이터들은 웹솔루션을 통해서 출력되는 재학증명서, 성적증명서, 재직증명서, 수강내역, 강의내역 등의 전자문서 형태로 확인한다. 그렇다고 이러한 전자문서들을 모두 파일이나 종이로 출력하여 보관하는 것은 현실적이지 않다. 대신에 데이터베이스, 웹서버, WAS, 웹솔루션 등 시스템 전체를 클라우드 컴퓨팅 환경으로 이전하여 에뮬레이션 하는 것이 좋은 해결책을 제공할 수 있을 것으로 기대된다.

세 번째, 청와대, 국민청원 등 국가적으로 중요한 홈페이지는 모습, 기능, 데이터 모두 원래의 모습 그대로 보존되어야 한다. 모습은 캡쳐화면, 데이터는 덤프파일 등으로 별도로 보관하는 것보다 에뮬레이션 전략을 도입하는 것이 모습, 기능, 데이터 모두를 장기보존하는데 좋은 해결책이 될 수 있다.

이렇게 에뮬레이션 전략은 기존의 마이그레이션 전략의 보완재 역할을 할 수 있으며, 클라우드 컴퓨팅 기술과 접목되어 에뮬레이션 전략이 도

http://ras.jams.or.kr http://dx.doi.org/10.14404/JKSARM.2019.19.4.001

입되면 로컬 환경에서 에뮬레이션 전략에 비해 아래와 같은 3가지의 큰 장점을 가지고 있다.

첫 번째로, 클라우드 컴퓨팅을 활용하는 근본적인 이유에서 장점을 확인할 수 있다. 클라우드 컴퓨팅 구축 허가에 대한 정책적인 결정만 이루어진다면 자체적으로 서버 구매하고 설치하는 것보다 빨리 시작할 수 있으며, 인프라 확장이 용이하고, 공간 및 유지관리 비용 절감할 수 있는 장점이 있다.

두 번째로, 클라우드 컴퓨팅 기술은 가상화, 분산처리 기술과 함께 발전해 왔다. 그러므로 클라우드 컴퓨팅을 구성하는 기술에는 기본적으로 다양한 운영 환경을 재생산하는 에뮬레이션 기술의 핵심이 기본적으로 포함되어 있다. 그러므로 클라우드 컴퓨팅 기술 이외에 에뮬레이션 기술이 별도로 신경 쓸 필요 없이 클라우드 컴퓨팅 기술에 지속적으로 관심을 가진다면 에뮬레이션 기술의 수준은 자연스럽고 높아진다.

세 번째로, 대국민 서비스에 용이하다. 클라우드 컴퓨팅 서비스는 기본적으로 원격에서 접속하는 것이 기본이기 때문이다. 본 연구의 클라우드 컴퓨팅 환경에서도 사용자가 웹브라우저를 통해서 OpenStackit에 접속하면 원하는 사양의 컴퓨터를 클라우드 컴퓨팅 환경에서 생성하여 접속할 수 있다. 이후, VNC, TeamViewer, 원격데스크탑 등 일반인에게도 익숙한 원격 접속 프로그램을 이용해서 시스템에 접근할 수 있다. 또한, 대국민 서비스를 위해서는 지속가능성이 중요하다. 클라우드 컴퓨팅 환경을 구성하는 OpenStack 구성 서버들이 공격당하면 서비스할 수 없다. 이러한 상황은 다른 서비스들에서도 마찬가지이므로, 클라우드 컴퓨팅 환경에서 특정 시스템을 에뮬레이션하는 가상서버에 국한한다. 해당 가상서버는 템플릿 이미지(설치파일)에서부터 생성되었고, 하나의 템플릿 이미지로부터 여러 개의 동일한 가상서버가 생성될 수 있다. 만약 하나의 가상서버가 외부로부터 공격받거나 이상 현상이 발생하여도 해당 가상서버를 클라우드 컴퓨팅 환경에서 삭제하고 다시 템플릿 이미지로부터 다른 가상서버를 생성한다면 서비스를 지속할 수 있다. 그러므로 클라우드 컴퓨팅 환경에서 에뮬레이션 전략을 활용한다면 기본적으로 서비스의 지속가능성이 높아진다.

3.7 클라우드 컴퓨팅 기반 에뮬레이션 전략의 한계점

IT기술은 항상 빠르게 발전하고 있으며 특히 시장 및 자본이 집중되는 분야에서는 더욱 그러하다. 클라우드 컴퓨팅 기술 CPU 프로세서 기술에 의존적이며, "현재 시점에서" 가장 많이 사용하고 있는 CPU 프로세서 기반으로 발전되어 있다. 현재 가장 많이 사용되고 있는 CPU 프로세서 구조는 Intel 및 AMD의 x86 CPU 프로세서이다. 클라우드 컴퓨팅 기술은 이들 CPU 프로세서를 중심으로 발전되었기 때문에 대부분의 가상화 기술은 x86 기반이다. 그래서, SPARC CPU 프로세서에 동작하는 HP-UX의 유닉스 계열 OS는 현재 원래 모습 그대로의 에뮬레이션은 어려운 실정이다. 현재 오래되었으면서 중요한 서버들의 OS에는 유닉스 계열이 많이 존재한다. 그러나 유닉스 계열의 OS는 현재 시장이 없어지고 있는 실정으로 IT 기술과 자본이 외면하고 있는 추세이다. 전자기록 장기보존에는 가장 발전되고 안정적인 IT 기술을 활용하여

야 하는데 IT 자본이 외면하고 있는 IT 기술이 성숙되기를 기다릴 수는 없다고 판단된다. 그러므로 유닉스 계열 OS의 경우에는 시스템이 보존되어야 할 필수 속성들을 도출하여 유닉스 계열 OS를 가장 유사한 리눅스 계열 OS로 전환하는 U2L(Unix-to-Linux)을 활용하여 원본과 동일 기능과 유사한 외관을 보여 줄 수 있는 방법이 좋은 해결책이 될 수 있다.

CPU, 메모리, 디스크, 네트워크 등 컴퓨팅 하드웨어 자원을 추상화하는 기술을 가상화 기술이며, 여러 서버들의 가상화된 컴퓨팅 하드웨어 자원을 하나의 컴퓨팅 자원 Pool로 묶고 기술을 분산처리 기술이다. 그리고 컴퓨팅 자원 Pool 상에서 사용자의 요청에 따라 알맞은 OS를 컴퓨팅 자원에서 설치하여 컴퓨터를 생성하는 것을 에뮬레이션 기술이라고 한다. 본 연구에서는 가상화, 분산처리, 에뮬레이션 기술로 각각 KVM, OpenStack, QEMU를 활용하고 있는데 각각의 기술은 지원하는 CPU 프로세스 종류, OS 등에 제한을 가지고 있다. 또한, 클라우드 컴퓨팅 환경에 신규로 서버를 생성하지 않고 기존의 레거시 서버를 클라우드 컴퓨팅 환경으로 전환할 때 활용되는 P2V 변환기인 zConverter도 지원하는 〈표 7〉과 같이 OS 종류에 제한이 있다. 이러한 기술들도 대부분의 Windows 계열과 리눅스 계열의 OS를 지원하고 있다. 그러므로 이들이 지원하지 않는 OS의 경우에는 지원하는 OS로 시스템 변환을 하는 U2L과 같은 방안이 효과적인 차선을 제공할 수 있다.

4. 맺음말

전자기록 장기보존의 핵심은 전자기록이 최초로 생성되고 활용되었던 본래의 기능적 속성과 비트스트림을 오랫동안 그대로 유지하는 것이다. 그러나 대부분 국내외 주요 아카이브 기관들은 경제적, 기술적인 측면을 고려하여 원본의 비트스트림에 변경을 허용하는 마이그레이션을 주전략으로 채택하고 있다. 그러나 전자기록 유형이 다양해지고 범위가 확장됨에 따라 마이그레이션 전략으로 진본성과 무결성을 유지하는데 한계를 드러나고 있다.

그러므로 본 연구에서는 클라우드 컴퓨팅 기술을 활용하여 전자기록의 비트스트림을 변경하지 않고 유지할 수 있도록, 전자기록이 생산·활용된 시스템 및 응용 환경을 재생산하는 에뮬

〈표 7〉 zConverter 지원 OS

구분	Windows	Linux
버전	Win 2012 R2 64bit Win 2008 R2 64bit Win 2008 ENT 64bit Win 2008 ENT 32bit Win 2003 64bit Win 2003 32bit	CentOS 4.8 32bit, CentOS 5.8 64bit, CentOS 6.3 32bit, CentOS 6.3 64bit, CentOS 6.4 32bit, CentOS 6.4 64bit, Ubuntu 10.04 32bit, Ubuntu 10.04 64bit, Ubuntu 12.04 32bit, Ubuntu 12.04 64bit SUSE Linux Enterprise 11 SP2 32bit SUSE Linux Enterprise 11 SP2 64bit

* 출처: Zconverter, http://www.zconverter.co.kr/index.php/product/Migration/

http://ras.jams.or.kr http://dx.doi.org/10.14404/JKSARM.2019.19.4.001

레이션 적용 가능성을 검토하고자 한다. 실제 에뮬레이션 테스트베드를 2개의 서버와 2개의 스위치로 구성하였으며, 하이퍼바이저는 KVM, 에뮬레이터는 QEMU를 사용하였으며, 이를 바탕으로 클라우드 컴퓨팅 환경을 구성하는 클라우드 운영체제는 OpenStack Queens 버전을 사용하였다.

네 가지 실험을 진행하였다. 첫 번째와 두 번째 실험은 재현하고자 하는 전자문서가 구동될 수 있는 환경을 클라우드 컴퓨팅 환경에 조성한 다음 전자문서를 재현하는 실험이었다. 연구개발 사업을 통해 클라우드 에뮬레이션 환경을 구축하여 검증한 결과, DOS 및 Windows XP환경에서 각각 보석글 워드프로세스와 한컴오피스 애플리케이션을 그대로 재현할 수 있었다. 이는 기록의 내용·구조·맥락에 더해 전자적 특성인 당시의 그대로의 외형과 기능을 재현함으로써 에뮬레이션이 기록의 속성을 보장하는 방안이 될 수 있음을 확인한 것이다. 그리고 세 번째와 네 번째 실험은 하나 이상의 서버들로 구성된 시스템 자체를 백업 도구를 활용하여 템플릿 이미지를 생성하여 클라우드 컴퓨팅 환경에 가상서버들을 설치하여 원본물리서버와 동일하게 실행하는 것을 실험하여 확인하였다. 이 실험을 통해서 전자기록의 유형이 다양한 형태로 확대되고 있으며, 디지털 객체는 물론 해당 객체를 구동하는 시스템 하드웨어 및 소프트웨어를 포함한 운영 환경까지 보존하는 것을 고려해야 하는 상황들이 생기고 있는 상황에서 시스템 전체를 이관하는 방식은 전자기록물을 장기보존할 수 있는 좋은 해결책 제시할 것으로 기대할 수 있었다. 또한, 클라우드 컴퓨팅 기반 에뮬레이션 전략은 클라우드 컴퓨팅 본래의 장점과 안정적으로 대국민 서비스를 운영할 수 있다는 장점도 있다.

그래서 에뮬레이션 전략은 마이그레이션 전략을 대체하는 방안이 아니고 현재 마이그레이션 전략을 해결하지 못하거나 포용하지 못하는 전자기록을 장기보존하는 보완 방안으로 활용할 수 있으며 오랫동안 확인하지 못했던 전자기록, 다양한 시스템 요소들이 연계되어 전자기록이 생성되는 경우, 국가적으로 중요한 홈페이지의 경우를 클라우드 컴퓨팅 기반 에뮬레이션 전략을 도입해야 한다.

그러나 현재 가장 x86 CPU 프로세스 기반으로 하는 클라우드 컴퓨팅 기술에 집중되어 있기 때문에 x86 이외에 다른 CPU 프로세스를 기반으로 OS는 직접 클라우드 컴퓨팅 환경에서 에뮬레이션 전략을 적용하는 것이 어렵다. 이러한 경우는 원본과 동일 기능과 유사한 외관을 보여줄 수 있으면서 클라우드 컴퓨팅 환경이 지원하는 OS로 변환하여 에뮬레이션 전략 도입하는 방법이 좋은 해결책이 될 수 있다.

지금까지 에뮬레이션은 마이그레이션 비해 비용이 높고 구현이 어려운 장기보존 전략으로 평가받아 왔다. 하지만 상용 에뮬레이터의 가격이 상대적으로 낮아지고 클라우드 컴퓨팅 기술이 발전하면서 가치 있는 중요 전자기록물에 대해 선별적으로 적용할 만한 기술로 진화되고 있다. 진화하는 디지털 환경에서 데이터세트, 트위터, 유튜브 등 국가 아카이브의 기록관리 대상은 급격히 증가하고 있다. 이에 대응하여 국가 아카이브 기술의 변화 추이를 지속적으로 모니터링하여 다양한 기록관리 전략을 확보할 필요가 있다.

http://ras.jams.or.kr http://dx.doi.org/10.14404/JKSARM.2019.19.4.001

참 고 문 헌

강맹수 (2019). 클라우드 컴퓨팅 시장 동향 및 향후 전망. 산은조사월보, 1(758), 54-71.

국가기록원 (2013). 행정기관 전자기록물 재현기술 연구 및 프로토타입 개발 완료보고서. 대전: 국가기록원.

김기정, 신동수 (2018). 클라우드 컴퓨팅 환경 영구기록물관리 시스템 구축 방안 연구. 한국기록관리학회지, 18(3), 49-70. DOI: https://doi.org/10.14404/JKSARM.2018.18.3.049

김명훈, 오명진, 이재홍, 임진희 (2013). 전자기록 장기보존 전략으로서의 에뮬레이션 사례 분석. 기록학연구, (38), 265-309.

김주영, 김순희 (2019). 클라우드 저장소를 활용하여 기록생산시스템에서 기록관리시스템으로 전자기록물을 이관하는 방안에 관한 연구. 한국기록관리학회지, 19(2), 1-24. DOI: http://doi.org/10.14404/JKSARM.2019.19.2.001

박종근, 최강일, 이상민, 이정희, 이범철 (2013). OpenStack 클라우드 네트워크 기술 분석. 전자통신동향분석, 28(5), 122-132.

소정의, 한희정, 양동민 (2018). 국외 전자기록물의 장기보존 정책 비교 분석. 한국기록관리학회지, 18(4), 125-148. DOI: http://doi.org/10.14404/JKSARM.2018.18.4.125

왕호성, 설문원 (2017) 행정정보 데이터세트 기록의 관리방안. 한국기록관리학회지, 17(3), 23-47. DOI: http://doi.org/10.14404/JKSARM.2017.17.3.023

이승억, 설문원 (2017). 전자기록관리정책의 재설계에 관한 연구. 기록학연구, (52), 5-37.

임지훈, 김은총, 방기영, 이유진, 김용 (2014). 클라우드 컴퓨팅 기반의 전자기록관리시스템 구축방안에 관한 연구. 한국기록관리학회지, 14(3), 153-179. DOI: http://doi.org/10.14404/JKSARM.2014.14.3.153

임진희 (2013). 전자기록관리론. 서울: 선인.

정예용, 심갑용, 김용 (2014). 클라우드 컴퓨팅 기반 중앙기록물관리시스템 설계 및 적용에 관한 연구. 한국비블리아학회지, 25(4), 209-233. DOI: http://doi.org/10.14699/kbiblia.2014.25.4.209

DPT (2006). 전자기록의 유형별 보존기법(=Digital Preservation Testbed. From digital volatility to digital permanence). (이미화, 현문수 공역). 서울: 한국국가기록연구원(원전 발행년 2003).

OLIVE (2019). Retrieved October 19, 2019, from https://olivearchive.org/

bwFLA (2019). Retrieved October 19, 2019, from http://eaas.uni-freiburg.de/

• 국문 참고자료의 영어 표기
(English translation / romanization of references originally written in Korean)

DPT (2006). From digital volatility to digital permanence(=Digital Preservation Testbed. From

digital volatility to digital permanence). (Lee, Mi Hwa & Hyun, Moon Soo). Seoul: The Research Institute for Korean Archives and Records.

Jung, Ye Yong, Shim, Gab Yong, & Kim, Yong (2014). A Study on Design and Application of Central Archives Management System Based on Cloud Computing. Journal of the Korean Biblia Society for Library and Information Science, 25(4), 209-233. DOI: http://doi.org/10.14699/kbiblia.2014.25.4.209

Kang, Maeng Soo (2019). Cloud Computing Market Trends and Future Challenges. KDB Monthly, 1(758), 54-71.

Kim, Ju young & Kim, Soon Hee (2019). A Study on Transferring Electronic Records from Record Production System to Record Management System Using Cloud Storage. Journal of Korean Society of Archives and Records Management, 19(2), 1-24. DOI: http://doi.org/10.14404/JKSARM.2019.19.2.001

Kim, Ki Jung & Shin, Dong Soo (2018). A Study on the Archives Management System in Cloud Computing. Journal of Korean Society of Archives and Records Management, 18(3), 49-70. DOI: https://doi.org/10.14404/JKSARM.2018.18.3.049

Kim, Myoung hun, Oh, Myung Jin, Lee, Jae Hong, & Yim, Jin Hee (2013). An Analysis of Cases of Emulation for Long Term Electronic Records Preservation Strategy. The Korean Journal of Archival Studies, (38), 265-309.

Lee, Seung Eok & Seol, Moon Won (2017). A Study of Redesigning Electronic Records Management Policies. The Korean Journal of Archival Studies, (52), 5-37.

Lim, Ji Hoon, Kim, Eun Chong, Bang, Ki Young, Lee, Yu Jin, & Kim, Yong (2014). An Application Method Study on the Electronic Records Management Systems based on Cloud Computing. Journal of Korean Society of Archives and Records Management, 14(3), 153-179. DOI: http://doi.org/10.14404/JKSARM.2014.14.3.153

National Archives of Korea (2013). Research and prototype development of electronic records reproduction technology. Daejeon: National Archives of Korea.

Park, Jong Keun, Choi, Kang Il, Lee, Sang Min, Lee, Jeong Hee, & Lee, Bum Chul (2013). Analysis of OpenStack Cloud Networking Technology. Electronics and telecommunications trends, 28(5), 122-132.

So, Jeong Eui, Han, Hui Jeong, & Yang, Dong Min (2018). A Comparative Analysis of Long-Term Preservation Policies in Foreign Electronic Records. Journal of Korean Society of Archives and Records Management, 18(4), 125-148. DOI: http://doi.org/10.14404/JKSARM.2018.18.4.125

Wang, Ho Sung & Seol, Moon Won (2017). A Study on Managing Dataset Records in Government Information Systems. Journal of Korean Society of Archives and Records Management, 17(3), 23-47. DOI: http://doi.org/10.14404/JKSARM.2017.17.3.023

Yim, Jin Hee (2013). Electronic Records Management. Seoul: Seon-in.

[첨부04] 용어정리

용어	설명
CDDL	Common Development and Distribution License(공동 개발 및 배포 허가서), 썬 마이크로시스템즈에서 만들었으며 MPL의 파생 라이선스
E-ARK 프로젝트	European Archival Records and Knowledge Preservation, 유럽의 전자정보 장기 보존을 위한 프로젝트
eCH 협회	스위스 전자 정부의 활성화를 위한 공공-민간 협력기구
GPL	General Public License, 자유 소프트웨어 재단(FSF)에서 만든 오픈 소스 소프트웨어를 위한 라이선스
LGPL	Lesser General Public License(약소 일반 공중 사용 허가서), GPL을 기반으로 라이브러리로서의 사용에 용이하도록 변경한 라이선스
OAIS 참조모형	전자 기록의 장기 보존을 위한 시스템의 개념적 기능틀을 제공하는 ISO 표준
SIARD 표준	SFA에서 관계형 데이터베이스에 저장되어 있는 데이터세트를 장기보존하기 위해 개발한 파일 포맷에 대한 표준
SIARD-DK	덴마크 시행령(bekendtgørelse) 1007/20(2010), 정보패키지에 대한 시행령으로서 전자기록입수에 관한 내용이 담겨있음
SQL:2008	SQL 데이터베이스 질의어를 위한 ISO (1987)와 ANSI (1986) 표준의 제6차 개정안
Table	DB에서 정보를 구분하여 저장하는 기본 단위로 Row, Column 등을 포함함
Row	Table의 행으로 값들의 나열을 뜻하며, 동의어로 Record가 있음
Column	Table의 열이며 동의어로 Attribute가 있음
Cell	Table의 행과 열이 교차하는 부분
SIARD 변환 (Download, 변환)	SIARD Suite의 기능을 제공받는 DBMS의 특정 Schema, Tablespace, DB 등을 SIARD 파일로 변환하는 행위
SIARD 복원 (Upload, 복원)	이용자가 SIARD Suite을 이용해 생성한 SIARD 파일을 DBMS의 DB 형태로 복원하는 행위
Key Type	변환 가능 검증 시험 항목 4가지 중 한 가지로 본 연구에서는 PK (Primary Key)와 FK (Foreign Key)를 통칭하는 말
Routine Type (루틴 타입)	변환 가능 검증 시험 항목 4가지 중 한 가지로 본 연구에서는 일련의 쿼리를 하나의 함수처럼 실행하기 위한 쿼리들의 집합인 Trigger, Function, Stored Procedures를 통칭하는 말
Primary Key(PK, 기본 키)	Table 내 Row를 구분하는데 기준이 되는 하나 혹은 그 이상의 Column의 집합인 후보 키 중 하나를 선정해 대표로 삼는 키
Foreign Key(FK, 외래 키)	다른 Table의 PK를 참조하는 것으로 Table의 관계를 나타내기 위해 사용하는 키
Open	SIARD Suite을 이용해 SIARD 파일의 data, column, user 등의 정보를 확인할 수 있도록 하는 것

Close	SIARD Suite에 Open 해놓은 SIARD 파일을 닫는 것
Metadata	SIARD 파일, Schema, Table, Column 등에 대한 메타데이터
클라우드 / 클라우드 컴퓨팅	인터넷을 통해 가상화된 컴퓨터의 시스템 리소스(IT 리소스)를 제공하는 것. 인터넷 기반 컴퓨팅의 일종으로 정보를 자산의 컴퓨터가 아닌 클라우드(인터넷)에 연결된 다른 컴퓨터로 처리하는 기술을 의미
가상화 / 가상화 기술	컴퓨터에서 컴퓨터 리소스의 추상화를 일컫는 용어. "물리적인 컴퓨터 리소스의 특징을 다른 시스템, 응용 프로그램, 최종 사용자들이 리소스와 상호 작용하는 방식으로부터 감추는 기술"로 정의
가상 머신 / 가상화 머신 / VM(Virtual Machine)	컴퓨팅 환경을 소프트웨어로 구현한 것, 컴퓨터를 에뮬레이션하는 소프트웨어. 가상머신 상에서 운영체제나 응용 프로그램을 설치 및 실행 할 수 있음
에뮬레이션	한 시스템에서 다른 시스템을 복제하는 것. 두 번째 시스템이 첫 번째 시스템을 따라 행동하는 것. 기록학에서 디지털 아카이빙의 보존 수단 중 하나. 디지털원본에 적용된 기술적인 조건들에 변경이 있어도 인코딩되어 있는 콘텐츠를 재생할 수 있는 환경을 프로그램으로 만들어내어 디지털정보의 접근성을 보장하는 기술.
운영 체제 / 오퍼레이팅 시스템 / OS(Operating System)	시스템 하드웨어를 관리할 뿐 아니라 응용 소프트웨어를 실행하기 위하여 하드웨어 추상화 플랫폼과 공통 시스템 서비스를 제공하는 시스템 소프트웨어를 말함. 마이크로소프트 윈도우, 맥 OS X, 리눅스, BSD, 유닉스 등이 있음
응용프로그램 / 애플리케이션	운영 체제에서 실행되는 모든 소프트웨어를 뜻함. 워드프로세서, 스프레드시트, 웹브라우저 등이 있음
하이퍼바이저(hypervisor)	호스트 컴퓨터에서 다수의 운영체제를 동시에 실행하기 위한 논리적 플랫폼을 말함. 가상화 머신 모니터 또는 가상화 머신 매니저라고 부름
마이크로소프트 윈도우	마이크로소프트가 개발한 컴퓨터 운영 체제. 현재 전 세계 90%의 개인용 컴퓨터에서 쓰고 있으며, 서버용 운영체제로도 사용되고 있음
리눅스	리누스 토르발스가 커뮤니티 주체로 개발한 컴퓨트 운영 체제. 자유 소프트웨어와 오픈소스 개발의 가장 유명한 표본. IBM, HP와 같은 거대 IT 기업의 후원을 받으며, 서버 분야에서 유닉스와 마이크로소프트 윈도우 운영체제의 대안으로 자리 잡음. 퍼블릭 클라우드 워크로드의 90%, 세계 스마트폰의 82%, 임베디드 기기의 62%, 슈퍼 컴퓨터 시장의 99%가 리눅스로 작동함
유닉스	교육 및 연구 기관에서 즐겨 사용되는 범용 다중 사용자 방식의 시분할 운영 체제. 리눅스와 오픈소스 BSD의 사용이 증가됨에 따라 기존의 상업 유닉스 시장이 침식되어 감
오픈소스	소프트웨어 혹은 하드웨어의 제작자의 권리를 지키면서 원시 코드를 누구나 열람할 수 있도록 한 소프트웨어 혹은 오픈소스 라이선스에 준하는 모든 통칭
오픈스택(Openstack)	IaaS 형태의 클라우드 컴퓨팅 오픈소스 프로젝트. 2012년 창설된 비영리 단체인 Openstack Foundation에서 유지, 보수하고 있으며 아파치 라이선스하에 배포 됨. AMD, 인텔, 레드햇, 델, HP, VM웨어 등 150개 이상의 회사가 이 프로젝트에 참여하고 있으며, 주로 리눅스 기반으로 운용과 개발이 이루어짐. 프로세싱, 저장공간, 네트워킹의 가용자원을 제어하는 목적의 여러 개의 하위 프로젝트로 이러져 있음
스냅샷	컴퓨터 파일 시스템에서 과거의 한 때 존재하고 유지시킨 컴퓨터 파일과 디렉터리의 모임. 가상화에서 스냅샷은 에뮬레이터가 게스트 운영 체제를 가상 머신에서 호스팅하고 컴퓨터의 상태 자체를 백업 파일로 옮김으로서 완전한 시스템을 복사하는 것을 말함
행정정보시스템	행정 정보를 통합 관리하여 여러 기관이 공동 이용함으로써 정책 결정 및 행정 처리의 신속화, 행정 자료의 중복 관리를 지양하고, 자료 관리의 표준화, 간소화로 자료의 활용도를 높이며 대민 봉사에 기여할 수 있는 시스템
데이터세트	컴퓨터가 처리하거나 분석할 수 있는 형태로 존재하는 관련 정보의 집합체. 데이터 파일이나 데이터베이스와 동의어로 사용됨. 정보가 컴퓨터로 처리하거나 분석할 수 있도록 구조화 되었음을 의미

주 의

1. 이 보고서는 국가기록원에서 시행한 용역연구개발사업의 연구결과 보고서입니다.

2. 이 보고서 내용을 발표할 때에는 반드시 국가기록원에서 시행한 용역연구개발사업의 연구결과임을 밝혀야 합니다.

3. 국가과학기술 기밀유지에 필요한 내용은 대외적으로 발표 또는 공개하여서는 아니 됩니다.