발간등록번호
11-1741050-100022-01

연구결과보고서

오픈 포맷을 활용한 행정정보 데이터세트 보존 방안 연구

A Study on Preservation Methods for Administrative Information Datasets Using Open Formats

주관연구기관 : 그리너리

국가기록원

※ 주 의 내 용

주 의

1. 이 보고서는 국가기록원에서 시행한 용역연구개발사업의 연구결과보고서입니다.

2. 이 보고서 내용을 발표할 때에는 반드시 국가기록원에서 시행한 용역연구개발사업의 연구결과임을 밝혀야 합니다.

3. 국가과학기술 기밀유지에 필요한 내용은 대외적으로 발표 또는 공개하여서는 아니 됩니다.

제 출 문

국가기록원장 귀 하

이 보고서를 “오픈 포맷을 활용한 행정정보 데이터세트 보존 방안 연구((주)그리너리/김병동)” 과제의 결과보고서로 제출합니다.

2025. 11. 28

주관연구기관명 : ㈜그리너리
주관연구책임자 : 김병동

공동연구기관명 : 전북대학교 산학협력단
공동연구책임자 : 양동민

목 차

연구결과보고서 요약문

연구과제명	오픈 포맷을 활용한 행정정보 데이터세트 보존 방안 연구		
중심단어	데이터세트, 보존포맷, 장기보존패키지, 전자기록, 기록물 재현		
주관연구기관	㈜그리너리	주관연구책임자	김병동
연구기간	2025. 4. 15. - 2025. 11. 30.		

○ 데이터세트 보존포맷 및 장기보존패키지 현황·분석

- 국내외 현황: InfoArchive(OAIS 기반), SIARD2·DBPTK(DB 보존), ETH Data Archive(오픈포맷), MARCXML·EDM(메타데이터 표준) 등 다양한 접근
- 문제점: DBMS 간 호환성 부족, BLOB/CLOB 한계, 비정형 데이터의 포맷 다양성, 대규모 저장 부담
- 대응방안: 국제표준 포맷(SIARD2, RDF/XML, PDF/A, TIFF, WAV) 채택, PID/SID 식별자 체계, 자동 변환·검증(DBPTK) 도입, 하이브리드 저장(S3+ 테이프)
- 기록 외 분야 사례: FAIR 원칙, 버전관리, 선택적 추출·포맷 마이그레이션 활용
- 법제도 분석: 미국 NARA·호주 PROV는 디지털화 시 원본 폐기 근거 명확. Migration은 명시 규정 부족하나 간접 규정 종합 분석 필요

○ 오픈포맷을 활용한 데이터세트 보존포맷 규격 설계

- 구조 설계: DB형 데이터세트의 구조적 특성을 반영하여 오픈포맷 기반의 보존포맷 설계 원칙과 구조를 제시
- 생성 절차 설계: 메타데이터, 데이터, 첨부파일, 재현파일, 무결성정보로 구성된 보존포맷의 생성 절차를 구체화함

○ 장기보존패키지(NEO3) 규격 설계

- 구조 정의: 「공공기록물법 시행령」·NAK 31-2 근거, 메타데이터·전자서명·원문 보존포맷 구성
- 보존 범위: DB 데이터+ 붙임은 필수, 재현 정보는 조건부, 주요 산출물은 제외 가능
- 메타데이터 설계: 국가기록원 SIP, 전북대 메타데이터(안) 비교·통합. NAK 8 준용하되 데이터세트 특성 반영 필요
- 기술정보 설계: OAIS 참조모형 기반, DC·EAD·MODS 매핑으로 상호운용성 확보
- 변경이력 관리·복원: 중복 최소화+ 무결성 확보 원칙. '최초 탐색 구성요소 복원 방식'으로 대용량 데이터도 완전 복원 가능

○ 테스트베드 구축 및 검증

- 검증용 DB 구축: 오아시스 3.0의 '개인인사기본'을 기반으로 16개 테이블과 493건의 데이터를 포함한 MySQL 테스트베드 구축
- 보존포맷 생성: 메타데이터·데이터·첨부파일 등 보존포맷을 구성하고 BLOB 변환 및 경로 이관 방식 정의
- 웹기록물 재현: XML-XSLT-HTML/CSS 기반 재현 프로세스를 확립하고 NEO3 구조에 통합
- 패키지 검증: 실데이터 검증으로 NEO3 패키지의 전자서명 진본성과 무결성 입증

Summary

Title of Project	A Study on Preservation Methods for Administrative Information Datasets Using Open Formats		
Key Words	Dataset, Preservation Format, Long-Term Preservation Package, Electronic Records, Record Reproduction		
Institute	GREENERY Co., Ltd.	Project Leader	Kim, Byung Dong
Project Period	2025. 4. 15. - 2025. 11. 30.		

❍ Dataset Preservation & Long-term Package (NEO3) Summary

- Current Approaches: Use of OAIS-based archives, DB preservation tools (SIARD2, DBPTK), open formats (ETH Data Archive), metadata standards (MARCXML, EDM)
- Challenges: DBMS incompatibility, BLOB/CLOB limits, diverse unstructured data, large-scale storage burden
- Strategies: Adopt standard formats (SIARD2, RDF/XML, PDF/A, TIFF, WAV), apply PID/SID identifiers, automate conversion/verification (DBPTK), hybrid storage (S3+ tape)
- Reference Practices: FAIR principles, version control, selective migration
- Legal Notes: U.S./Australia allow original disposal after digitization; migration rules unclear, requiring indirect legal analysis

❍ Design of Preservation Format for Dataset-type Records

- Structure Design: Presents design principles and structure for an open-format-based preservation format that reflects the structural characteristics of database-type datasets
- Creation Procedure Design: Specifies the detailed procedures for generating the preservation format, consisting of metadata, data, attachments, representation files, and integrity information

❍ Design of NEO3 Package for Dataset-type Records

- Structure: Based on law (Public Records Act, NAK standards), includes metadata, signatures, original formats
- Scope: Mandatory DB+attachments; conditional reproduction info; optional exclusion of outputs
- Metadata & Technical Design: Integrate existing SIP standards, apply OAIS model with DC/EAD/MODS mapping
- Integrity & Restoration: Ensure minimal redundancy and full data recovery via structured restoration methods

❍ Testbed & Verification

- Test Database Construction: Built a MySQL testbed based on the “Personal HR Information” menu of Oasis 3.0, including 16 tables and 493 records
- Preservation Format Creation: Structured the preservation format with metadata, data, and attachments, defining BLOB conversion and path migration methods
- Web Record Reproduction: Established an XML-XSLT-HTML/CSS-based reproduction process and integrated it into the NEO3 framework
- Package Verification: Verified the authenticity and integrity of the NEO3 package through real data validation and digital signature matching

총괄연구개발과제 연구결과

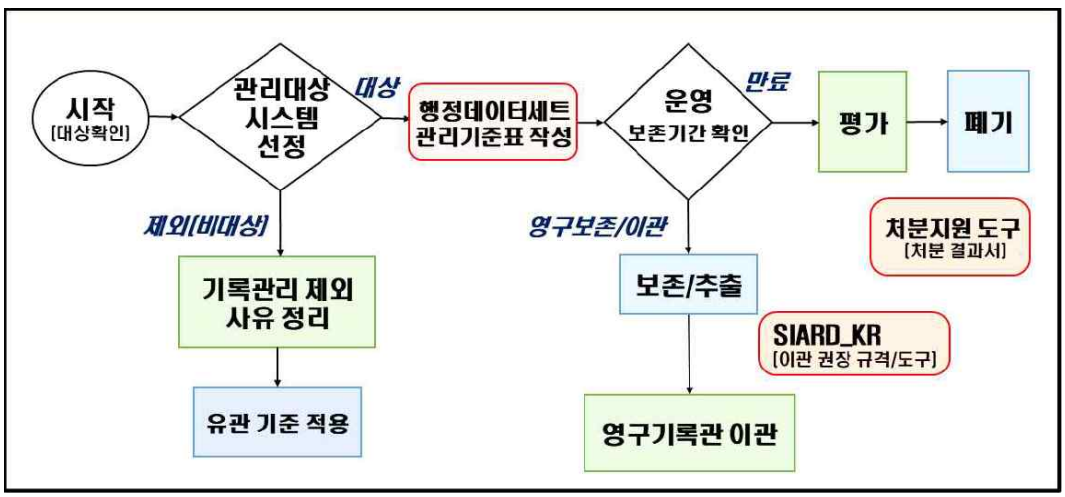
제1장 총괄연구개발과제의 최종 연구개발 목표

1.1 총괄연구개발과제의 목표

○ 국가기록원의 행정정보 데이터세트 기록관리 추진 현황

- 국내 행정정보 데이터세트(이하 데이터세트) 기록관리 방안은 다양한 공공기관 및 기록관리 주체들에게 필수적인 과제로 부상하고 있음. 국내에서는 행정안전부, 국가기록원, 중앙행정기관, 지방자치단체, 공공기관 등에서 행정정보시스템을 통해 생산·운영되는 데이터세트에 대한 기록관리 수요가 점차 확대되고 있음

- 국가기록원은 2017년 12월에 「행정정보 데이터세트 기록관리체계(안)」을 발표하였고, 2020년 3월에는 데이터세트에 대한 기록관리의 의무화 및 관리기준을 마련하기 위해「공공기록물 관리에 관한 법률 시행령」의 개정하였으며, 세부 실행방안을 위해 공공표준「행정정보 데이터세트 기록관리 기준(NAK 35:2020(v1.0))」과 기록관리 매뉴얼「행정정보 데이터세트 기록관리 실행 매뉴얼(기록관리 지침)」을 제정함. 이를 근거로, 각급 공공기관은 데이터세트 관리기준표를 작성 및 운영하는 기록관리업무를 이행해야 하는 법적 의무를 지니게 됨



<그림 1> 데이터세트 기록관리 절차

- 국가기록원은 데이터세트 관리기준표의 등록, 검색, 열람 등을 위한 관리 절차를 마련했으며 이관 대상 데이터세트를 SIARD 포맷으로 인수하여 보존·관리하는 종합관리시스템의 체계를 설계하고 구축함

- 또한 공공기관의 데이터세트 기록관리 업무를 조속히 안정적으로 정착시키고, 다양한 행정정보시스템에 대해 유연하고 효과적으로 대응할 수 있는 표준 및 기록관리지침을 마련하기 위해 2019년부터 2024년까지 「행정정보 데이터세트 기록관리 체계 구축」사업을 수행함

○ 데이터세트 이관 및 보존을 위한 보존포맷의 다양화 필요

- 현재, 국가기록원의 공공표준 「행정정보 데이터세트 기록관리 기준 - 관리기준표 작성 및 이관 규격」과 지침 「행정정보 데이터세트 기록관리 실행 매뉴얼」에서는 SIARD_KR(이관도구)를 통해 이관하는 것을 권장하고 있음

- SIARD_KR은 SFA와 E-ARK 프로젝트가 공동 개발한 SIARD 표준과 이관·복원·뷰어 기능을 제공하는 SIARD Suite를 국내 기록관리 환경에 맞게 개선한 이관 도구임. 테이블 단위 이관·복원, 첨부파일 포함, CUBRID 지원 기능 등을 추가하여 국내 활용 가능성을 높였음

- 그러나 데이터세트의 보존포맷으로 활용되고 있는 SIARD_KR은 이관 대상 DBMS의 종류와 개수가 제한적이며, 공개된 버전 기준으로만 적용할 수 있다는 한계가 있음. 또한 기존에 지원하던 DBMS조차도 버전 업그레이드 등의 변화가 발생하면서 SIARD_KR에 반영되지 않는 경우가 많아, 실제 이관 과정에서 SIARD_KR이 완벽하게 지원하지 못하는 사례가 다수 발생하고 있음

- 또한 SIARD는 테이블 단위의 이관은 가능하지만, 열이나 행 수준의 세부 선택이 불가능해 불필요한 데이터까지 포함되는 한계가 있음. 실제 사용자 화면을 재현하려면 웹서버나 WAS 등 추가적인 환경 구현이 필요하다는 점도 구조적인 제약으로 지적됨. 이에 따라 SIARD의 기능 개선이 요구되고 있으나, 지속적인 인적·물적 자원이 필요하다는 점에서 현실적인 어려움이 존재함

- 최근 SIARD 외에 새로운 데이터세트 보존포맷에 대한 수요가 점차 증가하고 있음. 현재 관계형 데이터베이스 기반 보존포맷은 사실상 SIARD가 유일하기 때문에, 국내 기록관리 환경에 적합한 새로운 포맷 규격을 마련하는 것이 보다 현실적임. 이때 개발되는 보존포맷은 장기적인 열람과 활용을 위해 개방형 국제표준이나 업계에서 널리 쓰이는 사실상(De Facto) 표준 기반으로 설계되어야 함

➔ **데이터세트 이관 및 보존을 위해 SIARD 이외에, 널리 활용되고 국제 표준포맷으로 제정된 오픈 포맷을 활용한 데이터세트 보존포맷 다양화가 반드시 이루어져야 함**

○ 데이터세트에 적합한 장기보존패키지(NEO3) 규격 마련 필요

- 2008년 국가기록원은 전자기록의 진본성을 보장하고 장기적으로 보존하기 위해 ISO 14721 OAIS 참조모형에서 제시한 정보패키지 구조와 내용을 기반으로 보존체계를 설계함. 이 과정에서 호주 PROV(Public Record Office Victoria)의 AIP(Archival Encapsulated Object)인 VEO를 벤치마킹하고, Base64 인코딩 방식을 적용한 정보패키지 형태의 NEO 표준을 개발함. 이후 NEO는 전자기록물의 장기보존포맷으로서 관련 시행령에 명시되어 공식적으로 채택됨

- 전자기록물의 용량 증가와 유형 다양화에 따라, 크기가 커지고 생성 시간이 오래 걸리는 Base64 인코딩 기반의 NEO 포맷은 ZIP 기반으로 개선됨. 이에 따라 기존 패키징 방식은 'NEO2', 개선된 방식은 'NEO3'로 명칭을 구분하고, 공공표준 NAK 31-1과 NAK 31-2로 개정·제정함

- NEO3는 문서 유형을 포함하여 비디오, 오디오, 이미지, 데이터세트 등 다양한 전자기록물 유형을 수용할 수 있도록 설계되었지만, 기존의 문서 중심 보존포맷인 NEO2를 기반으로 개발되었기 때문에 여전히 문서 유형에 특화된 구성 요소와 정의가 일부 남아 있음. 또한 NEO3는 현재까지 문서 유형 이외의 전자기록 유형에 실제로 적용된 사례가 없어, 다양한 유형의 기록물에 곧바로 적용하기에는 제한이 따르는 상황임

- 특히, NEO3 장기보존패키지의 구성요소인 원문, 보존포맷, 장기보존 메타데이터, 진본확인 정보에 어떤 내용을 포함해야 하는지에 대한 구체적인 기준이 부족함. 원문 개체의 설정 기준이 명확하지 않고 데이터세트에 특화된 장기보존 메타데이터 정의도 마련되어 있지 않아, 이에 대한 신규 정의가 필요한 상황임

➔ **따라서, 현재의 장기보존패키지 NEO3가 데이터세트 유형의 전자기록물을 수용할 수 있도록 규격을 전면 개정하거나 데이터세트 유형에 적합한 새로운 장기보존패키지 규격을 제정할 필요가 있음**

1.2 총괄연구개발과제의 목표달성도

구분	계획대비 달성도 (※ 8월 기준)	전체 달성도 (※ 11월 기준)	달성 내용	추후계획
문헌사례 분석 (그리너리/ 전북대)	100%	100%	· 국내외 아카이브 기관 및 기록관리 외 분야를 대상으로 데이터세트 보존포맷 및 장기보존패키지 현황 조사 및 분석 · 디지털화 또는 마이그레이션 후 원문 폐기와 관련된 해외 법령 및 표준 조사·분석 · 디지털화·마이그레이션·데이터세트 기록관리 관련 해외 사례 조사(호주 PROV 서면 질의·회신 결과 정리)	-
보존포맷 설계 (전북대)	95%	95%	· SIARD표준 및 SIARD_KR 분석 · XSD/XML 기반 표준화된 파일 포맷 분석 · DB형 데이터세트 유형 오픈포맷 기반 보존포맷 설계	-
보존포맷 검증 (그리너리)	95%	95%	· 보존포맷검증 대상 정보시스템의 업무기능, 데이터세트 구성, 사용자 화면(GUI) 설정 · 보존포맷 설계 방안에 따른 다양한 DBMS(MySQL, MS SQL Server, Oracle)에 대한 실데이터 기반 검증	-
NEO3설계 (전북대)	100%	100%	· 장기보존패키지(NEO3) 구조 및 구성요소 정의 · 데이터세트 유형에 적합한 장기보존 메타데이터 및 기술정보 설계 · 장기보존패키지(NEO3) 생성 및 이력 관리 방안 설계 · 재현 관련 표현 정보, 산출물의 기록물 정의 포함 여부에 따른 다양한 시나리오 도출	-
NEO3검증 (그리너리)	100%	100%	· 장기보존패키지(NEO3)의 검증을 위한 시나리오 수립 · 장기보존패키지(NEO3) 검증을 위한 테스트베드 구축 · 실데이터 기반 장기보존패키지(NEO3) 검증 수행 · 장기보존패키지(NEO3)검증 체계와 보존포맷 검증 간 연계 방안 도출	-

1.3 국내·외 기술개발 현황

가. 국외 - 데이터세트 유형 보존포맷 및 재현 방안

○ 미국 NARA(National Archives & Records Administration)

- NARA는 다양한 유형의 전자기록물을 장기보존하기 위해 마이그레이션을 보존 전략으로 채택하고, 보존을 위한 포맷 유형을 선호포맷, 허용포맷, 즉시 이관 허용(imminent transfer acceptable) 포맷으로 구분하여 제시함

- NARA는 자체적으로 수립한 위험 매트릭스(risk matrix)를 활용하여 보유 디지털 기록물의 파일 포맷 보존 위험도 평가하고, 상대적 위험 수준에 따라 변환이 필요한 파일포맷을 식별함. 현재 740여 개의 파일포맷 버전을 관리하고 있음

- NARA는 디지털 기록을 문서, 프리젠테이션, 데이터세트, 이미지, 오디오, 동영상, CAD, 지리공간, 웹, 이메일 등 총 16개 유형으로 구분하고, 유형별 보존 실행 계획을 수립하여 필수보존속성을 정의함

- 데이터세트(Structured Data)는 캘린더(Calendars), 데이터베이스(Databases), 스프레드시트(Spreadsheets) 총 3가지로 분류하여 각각의 보존 실행 계획을 적용해 관리함

○ 영국 TNA(The National Archives)

- TNA는 이관된 디지털 기록을 원본 포맷 그대로 보존하는 것을 원칙으로 함. 원본 비트스트림을 유지할 수 있도록 에뮬레이션을 기본 보존 전략으로 채택하며, 보존 위험을 관리할 방법이 없는 경우에 한해서 마이그레이션을 예외적으로 허용함

- 디지털 기록 보존 및 포맷 위험 관리를 위해 DROID, PRONOM 등 디지털 파일포맷 식별 도구 및 포맷 정보 데이터베이스를 자체 개발하여 사용함

- TNA는 홈페이지에서 '이관을 위한 파일포맷(File formats for transfer)'을 유형별로 제공하여 장기보존이 가능한 파일포맷의 범위를 안내함. 목록에 포함되지 않은 파일포맷도 이관할 수 있으나 추가 검토가 필요할 수 있으므로 사전 협의가 필요함. 관련 파일포맷명과 확장자 등의 정보를 주기적으로 업데이트하여 제공함

- 이관 파일포맷 목록에는 데이터세트 유형이 포함되지 않았으나 TNA는 보존하고 공개할 디지털 기록 유형 중 데이터세트와 데이터베이스가 있음을 언급함. 또한 PRONOM에서 이에 해당하는 파일포맷을 검색하여 관련 정보를 확인할 수 있음

○ 호주 NAA(National Archives of Australia)

- NAA는 디지털 기록의 원본 포맷과 비트스트림을 유지하는 것을 원칙으로 하되, 파일 포맷의 위험도를 지속적으로 모니터링하여 보존에 취약한 포맷은 선호 포맷으로 마이그레이션하는 보존 전략을 적용함

- 보존 대상이 되는 디지털 기록은 디지털로 생산된 기록(born-digital records)과 디지털화 기록으로 구분된다. 이 중 디지털로 생산된 기록은 업무문서, 이메일, 정지 이미지, 오디오, 시네마, 비디오, 비디오 컨테이너, CAD, 지리 공간, 구조화 데이터, 웹 사이트 등으로 분류됨

- NAA는 디지털로 생산된 기록의 파일포맷 표준을 통해 장기보존이 가능한 파일포맷을 규정함. 해당 표준은 국제 모범 사례를 기반으로 하며, 호주 정부 기관에서 일반적으로 사용하는 포맷과 보존에 적합한 추가 파일포맷을 포함하도록 정기적으로 검토 및 업데이트됨

- NAA는 데이터세트를 구조화 데이터(structured data)로 분류하며, 장기보존에 적합한 선호포맷으로 csv, ebcdic, xml, json, tsv;tab를 제시함

-

○ 스위스 SFA(Swiss Federal Archives)

- SFA는 마이그레이션을 핵심 보존 전략으로 적용함. 디지털 환경 변화에 맞춰 필요시 새로운 보존 가능한 포맷으로 변환하며, 변환과정은 문서화하여 추적이 가능하도록 함. 보존을 위한 포맷 유형은 '장기보존 파일포맷(archivable file formats)'으로 규정되며, 제출된 파일이 기준을 충족하지 않을 경우 적절한 포맷으로 변환 후 보존함

- 보존 포맷의 위험도를 정기적으로 평가하고, 저장된 디지털 기록물의 무결성과 접근 가능성을 모니터링함. 보존 형식의 적절성을 유지하기 위해 파일포맷, 저장 매체, 소프트웨어 등을 지속적으로 점검하며, 필요한 경우 새로운 포맷을 도입하여 기존 데이터를 변환함

- OAIS 모델을 기반으로 보존체계를 운영하며, 데이터 저장 및 접근 체계를 표준화함

- 데이터세트는 Tables and Databases 유형으로 분류됨. 단순한 테이블 데이터는 CSV 포맷으로, 복잡한 관계형 데이터베이스는 SIARD 포맷을 사용하여 보존함

- SFA가 제안한 SIARD(Software Independent Archiving of Relational Databases)는 관계형 데이터베이스 구조를 유지하면서도 장기보존이 가능하도록 설계된 파일포맷임. OAIS 패키지 모델과는 독립적으로 설계되었으며, 자체 메타데이터를 포함하고 관련 문서들과 함께 보존되는 것을 전제로 함

  · SIARD 2.1은 SQL:2008 표준을 기반으로 메타데이터와 테이블 데이터를 XML 형식으로 저장하며, 모든 데이터를 하나의 SIARD 파일 내에 포함되고 ZIP 형식으로 압축하여 보관함

  · SIARD 2.2는 대용량 객체(Large Object, LOB) 데이터를 SIARD 아카이브 외부 저장소에 보관하는 기능을 지원하며, 외부 파일은 "file:" URI를 통해 참조됨

  · SIARD Suite는 SIARD 파일을 데이터세트 형태로 열람할 수 있도록 지원하는 전용 뷰어로, 저장된 데이터의 조회 및 검색, 메타데이터 비교· 수정, 데이터 구조 유지 등을 지원함

나. 국내 - 데이터세트 유형 보존포맷 및 재현 방안

○ 국가기록원 NAK(National Archives of Korea)의 데이터세트 보존포맷 및 재현관련 추진 사항

- 2022년 이전(『공공기록물법 시행령』 2022.07.05. 개정 전)

  · (단일 포맷 전략의 한계) 국내 전자문서 보존 정책에 따라 보존기간이 10년 이상인 전자문서를 문서보존포맷(PDF/A-1b)으로 변환하고 원본과 함께 장기보존포맷(NEO2)로 패키징하여 관리해왔음. 그러나 PDF/A-1b는 시각적 형식 보존에 적합하지만, 그 외 기록물의 중요 속성을 유지하기 어렵다는 한계가 있음

  · (전자기록 유형별 포맷 정책 방향 제시) 기록물 생산환경과 기술이 발전함에 따라 전자문서 외에도 데이터세트, 시청각기록물, 웹기록물 등 다양한 유형의 전자기록이 생산되고 있음. 그러나 단일 문서보존포맷으로는 이러한 기록물을 효과적으로 수용하기 어려운 한계가 있음. 이에 기록물 유형별 보존포맷을 다양화하고, 장기보존 정책의 유연성을 확보하기 위한 연구가 수행됨

  · (데이터세트 장기보존 및 활용 관점에서의 SIARD 도입 방안 제시) 국가기록원은「데이터세트 유형 전자기록의 장기보존기술 연구」를 통해 데이터세트 보존을 위한 마이그레이션 전략 수립, 문서보존포맷 검증, 에뮬레이션 적용 가능성 평가 등 기술적 연구를 수행함. 특히 이 연

구에서는 데이터세트 보존포맷으로 가장 많이 논의되는 SIARD 표준을 대상으로 적합성 평가 및 기술적 검증을 함. 모든 기록유형에 적용할 수 있는 공통기준과 데이터세트에 특화된 고유기준으로 구성된 평가체계를 개발하고, 이를 적용하여 SIARD의 보존포맷 적합성을 평가함. 데이터세트는 시각적 표현보다는 데이터의 콘텐츠와 기능이 DBMS에 재현될 수 있는지 그 여부가 보존의 핵심임. 이에 따라 다양한 DBMS에서 데이터를 SIARD 파일로 변환하고, 이를 다시 DBMS에 업로드한 뒤 데이터의 정합성을 검증함. 또한 SIARD의 단점을 보완하기 위해 국산 DBMS인 큐브리드를 대상으로 SIARD Suite의 기능을 확장·개발하고, SIARD 포맷의 안정성 향상을 위해 변환 불가능 항목 처리 기능과 변환 중 일시 정지 기능을 추가하는 등 실용적 개선 방안을 도출함

· (데이터세트 기록관리 지원도구(SIARD_KR) 고도화) 국가기록원은 2020년 행정정보 데이터세트 기록관리 체계구축사업에서 SIARD Suite를 기반으로 국내 기록관리 환경에 맞게 SIARD_KR을 개발함. 이 도구는 SQL:2008, XML, ZIP64, JDBC API 4.1 등 다양한 기술 표준을 활용하여 DBMS에서 추출된 데이터와 구조를 XML 형식으로 변환하고, 이를 .siard 형식으로 저장함. SIARD_KR은 기존 기능에 더해 국산 DBMS인 큐브리드 지원, BLOB 이외의 첨부파일 추출 기능, DB 전체뿐만 아니라 테이블 단위 추출 기능이 추가되어 고도화됨

· (SIARD 이관 권고) 국가기록원은「행정정보 데이터세트 기록관리 기준」표준을 제정하고, 해당 표준에서 SIARD 형식으로의 이관을 권고함. 다만, 이는 의무사항은 아니며 권고사항으로 제시됨

· (웹 기반 SIARD 지원도구 프로토타입 SW 개발) 국가기록원은「행정정보 데이터세트 기록정보 서비스 및 활용 모형 연구」를 통해 영구기록물관리기관으로 이관된 데이터세트(SIARD 파일)의 대국민 서비스 제공을 위한 기반 기술로서 웹 기반 SIARD_KR 지원도구 프로토타입을 개발·배포함. 해당 도구는 사용자가 웹 인터페이스를 통해 DB 접속정보 및 파일명을 입력하면 SIARD 파일로 변환(다운로드)하거나, SIARD 파일을 다시 DB로 복원(업로드)할 수 있도록 구현됨. 또한 테이블 형태의 실제 데이터 내용을 확인할 수 있는 기능도 제공함. 본 시스템은 향후 데이터세트의 장기보존과 활용을 위한 통합관리체계의 기반 인프라로 활용될 수 있음

- 2022년 이후(『공공기록물법 시행령』2022.07.05. 개정 후)

· (『공공기록물법 시행령』2022.07.05. 개정) 문서보존포맷을 보존포맷으로 용어를 변경하면서 다양한 유형의 전자기록으로 보존포맷을 확대 및 다변화할 수 있는 기반을 마련함. 장기보존포맷은 장기보존패키지로 본래의 의미를 반영하여 용어를 변경하였으며, 이로 인해 보존포맷과 개념적으로 분리됨

· (「전자기록물 보존포맷 선정기준」 공공표준 제정) 모든 유형의 전자기록물에 적용할 수 있는 보존포맷 선정절차와 방법을 체계화함. 현재 공통기준과 문서 유형(텍스트, 스프레드시트, 프리젠테이션)에 대한 고유기준을 제시하고 있음

· (전자기록 유형별 고유기준의 다양화) 현재 보존포맷 선정체계는 문서 유형에 국한되어 있으며, 다양한 전자기록에 대해 보존포맷을 확대하고 다양화하는 것이 필요함

· (데이터세트 유형 보존포맷 선정기준 개발) 국가기록원은 데이터세트 유형의 장기보존을 위해 보존포맷 선정기준의 체계화를 추진함. 빅데이터 환경에서 데이터 관리의 중요성이 두드러지면서 데이터세트의 보존 적합성을 평가할 수 있는 고유기준 마련 필요성이 제기됨. 이에 따라 데이터세트의 필수보존속성을 도출하고, 이를 바탕으로 보존포맷 선정 고유기준을 설계함. 데이터세트는 데이터베이스형과 구조화데이터형으로 구분하여 각각의 고유기준을 수립함. 이 기준은 데이터세트 기록의 체계적 장기보존을 위한 실질적 기반으로 평가됨

· (행정정보 데이터세트의 재현 전략 고도화 방향 제시) 국가기록원은 데이터세트의 지속적 보존과 활용을 위해 SIARD 방식 외에도 구조화 데이터(XML, CSV, JSON 등)와 가상화 솔루션(OVF/OVA)을 포함한 세 가지 이관·보존 방안을 제안함. 특히 XML/XSLT 기반 이관 방

식을 시범기관에 적용하여 실제 이관 및 재현 과정을 수행함으로써, 데이터세트 유형에 적합한 보존 포맷과 재현 방식의 타당성을 검증함. 구조화 데이터 포맷은 데이터세트 단위 이관에 적합하며, 시스템 전체 기능의 재현이 필요한 경우에는 가상화 포맷의 활용이 바람직함. 또한 질의 결과 화면 등 사용자 인터페이스 재현에는 XML/XSLT 기반 방식이 유효한 대안으로 제시됨. 아울러 재현은 기관의 환경, 자원, 시스템 특성에 따라 유연하게 결정되어야 하며, 사용자 인터페이스 역시 데이터세트의 필수보존속성에 포함될 수 있음을 확인하고, 이를 반영한 재현 전략의 필요성이 강조됨. 본 연구는 다양한 이관 포맷과 재현 방식의 적용 가능성을 검토함으로써, 데이터세트 장기보존 정책의 실효성을 제고하고 향후 체계적인 재현 모델 수립을 위한 기반을 마련함

○ 국가철도공단의 데이터세트 보존포맷 및 재현관련 추진 사항

- 국가철도공단은 데이터세트 전자기록의 체계적 관리와 장기보존을 위해 다양한 기술 기반을 마련하고 있음. 데이터 기반 기록정보자원의 효율적 관리와 핵심 정보자산의 장기적 활용 가치를 제고하기 위해, 데이터세트 전자기록 관리기준을 수립하고 차세대 기록관리시스템(ARMS)에 데이터세트를 이관하고 있음. 특히, 이관 대상 데이터세트의 주요 속성정보를 포함한 표준화된 패키지를 생성하기 위해 범용 입수규격생성기(SIP Creator)를 개발하였으며, 이를 통해 연계 표준을 반영한 다양한 유형의 데이터세트 전자기록을 체계적으로 관리하고 장기보존할 수 있는 기반을 마련함. 또한 공단은 표준전자문서뿐 아니라 다양한 디지털 객체를 포괄할 수 있는 장기보존 패키지로 BagIt을 채택함. BagIt 포맷은 단순하면서도 유연한 구조를 통해 콘텐츠와 설명 정보를 함께 포장할 수 있으며, 이관, 무결성 검증 등 다양한 기록관리 기능에 활용 가능하여 데이터세트의 장기보존에 적합한 포맷으로 평가됨. 공단은 이러한 기술적 시도와 지속적인 연구를 통해 국가기록원의 정책 방향에 부합하는 선진 보존체계를 제시하고, 공공기록관리의 고도화를 선도하고자 함

다. 국외 - 장기보존패키지 현황

○ 미국의 BagIt (의회도서관)

- BagIt은 디지털 콘텐츠의 저장 및 전송을 위한 계층적 파일 시스템 구조를 정의하는 규칙 집합으로, 임의의 디지털 자산을 안정적으로 보존하고 공유하는 데 활용됨. ‘bag’은 ‘payload(digital files)’와 bag의 저장 및 전송을 문서화하기 위한 메타데이터 파일인 ‘tags’로 구성됨. 필수 태그 파일에는 payload의 모든 파일과 해당 체크섬을 나열하는 manifest가 포함됨

- 2008년 John Kunze에 의해 IETF(Internet Engineering Task Force) 초안으로 제안되었으며, 이후 여러 차례 개정을 거쳐 2018년 IETF에서 버전 1.0이 RFC로 공식 채택됨. 현재 BagIt은 미국 의회도서관(The Library of Congress), Dryad Data Repository, NSF DataONE, Rockefeller Archive Center를 포함한 다양한 기관에서 도입하고 있으며, 코넬, 퍼듀, 스탠퍼드, 겐트, 뉴욕, 캘리포니아 대학교 등 주요 대학 도서관에서도 활용되고 있음

- 또한 Python, Ruby, Java, Perl, PHP 등 다양한 프로그래밍 언어 기반의 소프트웨어 구현이 가능해, 다양한 환경에서 유연하게 적용할 수 있는 포맷으로 평가됨

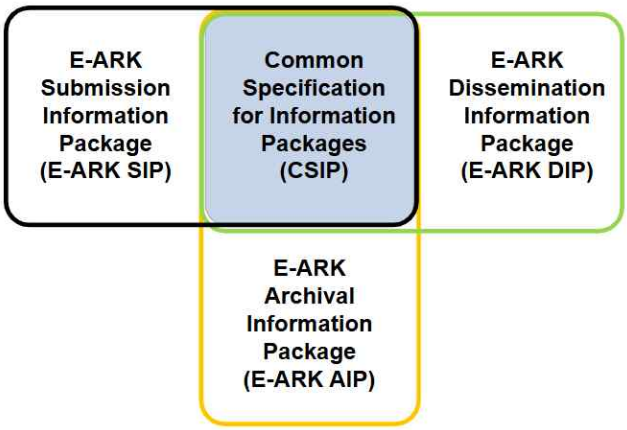
○ 유럽의 E-ARK 정보패키지(CSIP, SIP, DIP, AIP)

- ISO 14721:2012 - OAIS 참조모델에서 정의하는 정보패키지의 구조에 대한 자세한 지침이 부재함. 이러한 간극을 해결하기 위해 EU의 지원을 받은 E-ARK(European Archival Records and Knowledge Preservation) 프로젝트가 표준화된 패키지 사양의 개발을 시작함

- E-ARK는 상호운용성을 위한 IP 공통 명세서인 E-ARK CSIP 및 E-ARK SIP·AIP·DIP 등의 데이터 패키징 기술 명세서를 제공함. 아래 <그림 2>로부터 정보패키지와 공통 명세서 간의 관계 구조를 파악할 수 있음

- CSIP는 모든 정보패키지에 대한 공통 구조 및 핵심 메타데이터 요구사항을 정의함. 즉, 가장 중요하고 일반적인 정보패키지 상호운용성 측면을 정의함

- 또한, E-ARK SIP, E-ARK AIP 및 E-ARK DIP에 대한 확장 구현 사양이 함께 제공됨. 일반적으로 CSIP에 제시된 요구사항을 기반으로 구축되며, 특정 프로세스에서 추가 요구사항으로 확장됨



<그림 2> OAIS 정보패키지와 관련된 CSIP의 범위

- E-ARK SIP, E-ARK AIP 및 E-ARK DIP는 하나 이상의 콘텐츠 유형(관계형 데이터베이스, 지리 공간 데이터, 3D 모델, 소프트웨어 코드 등)으로 콘텐츠를 보유할 수 있음

- E-ARK는 OAIS 참조모델, PAIS(Producer-Archive Interface Specification) - CCSDS 등을 기반으로 함

- 2017년 2월 17일 버전 0.1로 시작하여, 2024년 5월 17일 버전 2.2.0으로 개정됨

❍ 호주의 PROV

- 호주 빅토리아주의 보존 기록관인 PROV(Public Record Office Victoria)는 디지털 기록물의 장기 보존을 위해 VEO(VERS Encapsulated Object)로 변환하여 보존할 것을 권장하고 있음. VEO는 기록물과 메타데이터를 캡슐화하고, 무결성을 보장하기 위해 디지털 서명 기술을 사용하여 서명됨

- VEO에 포함된 기록물은 승인된 장기보존포맷으로 캡슐화되며 VEO에는 기록에 대한 맥락 정보를 포함하는 메타데이터 패키지가 들어있음

- PROV는 VEO2로 캡슐화한 디지털 기록물 이관을 허용하지만, 가능한 경우 VEO3를 권장하고 있음

- VEO3는 VEO2에 비하여 메타데이터 요구사항의 엄격함이 보다 낮고, 다중 레벨과 계층적 폴더 구조의 기록물을 지원하여 VEO2보다 유연함

- 상용제품 혹은 수동 프로세스를 통해 VEO3를 생성하는 경우, 표준 및 지침의 요구사항을 충족하는지 확인하기 위해 VEO 검증프로그램을 통한 검증이 필요함

- 현재 VEO에 대한 모든 참조는 VEO3를 의미함

라. 국내 - 장기보존패키지 현황

○ 국가기록원 NAK(National Archives of Korea)의 데이터세트 장기보존포맷 관련 추진 사항

- 2022년 이전(『공공기록물법 시행령』 2022.07.05. 개정 전)

· (전자문서 중심의 기록관리) 국가기록원(중앙기록물관리기관)이 주도하고 있는 국내 기록관리 체계는 전자문서 중심으로 설계되어 수행되고 있음. 국내 기록관리의 가장 큰 원칙은 공공기관에서 생산되는 모든 전자문서를 보존하는 것이며, 그 기간은 각 전자문서에 설정된 보존기간(7종)으로 결정됨

> 기록물의 보존기간 7종은 기록물의 보존가치에 따라
> 영구, 준영구, 30년, 10년, 5년, 3년, 1년으로 구분하며 보존기간이 만료되면 폐기한다.

· (단일 포맷 전략) 전자문서에 대한 보존 정책의 핵심 전략은 장기보존포맷(NEO2)과 문서보존포맷(PDF/A-1)임. 따라서 보존기간이 10년 이상으로 책정된 전자문서는 PDF/A-1로 변환되어 원본과 함께 NEO2 포맷으로 패키징 되어 관리됨. 원문의 보존포맷 그리고 패키징 모두 한 가지 방식으로만 변환되고 관리되는 상황임

> 장기보존포맷(NAK's Encapsulated Object, NEO)는
> 전자기록 원문, 문서보존포맷, 메타데이터, 전자서명을 XML을 이용하여 하나의 정보패키지로 묶은 파일이다.

· (단일 포맷 전략의 한계 인식) 보존포맷은 출력 가능한(printable) 파일포맷인 PDF/A-1가 문서용으로만 정해져 있어서 엑셀 등의 다른 파일포맷에 대한 대응이 미비함. 게다가 공공기관에서 주요 업무과정에서 생성되는 데이터세트, 기관의 대표적인 정보가 게시되는 웹기록, 대부분 대용량으로 생성되는 동영상, 특정 소프트웨어로만 구동되는 CAD 파일 등 다양한 유형의 디지털 파일을 장기보존하기 위한 포맷 정책 또한 다변화할 필요가 있음

· (전자기록 유형별 포맷 정책 방향 제시) 기록물 생산환경과 기술의 변화로 다양한 유형의 전자기록이 생산되고 있으나 현행 단일 문서보존포맷으로 대응에 한계가 있음. 이를 극복하기 위해서 전자기록 장기보존 정책의 방향을 제시함

- 2022년 이후(『공공기록물법 시행령』 2022.07.05. 개정 후)

· (『공공기록물법 시행령』 2022.07.05. 개정) 문서보존포맷을 보존포맷으로 용어를 변경하면서 다양한 유형의 전자기록으로 보존포맷을 확대 및 다변화할 수 있는 기반을 마련함. 장기보존포맷은 장기보존패키지로 본래의 의미를 반영하여 용어를 변경하였으며, 이로 인해 보존포맷과 개념적으로 분리됨

· (장기보존패키지(NEO3) 공공표준 제정) 대용량 기록의 경우 향후 더욱 많은 생산이 예상되므로 변환시간과 저장공간을 줄여 패키징할 수 있는 NEO3 방식을 추가로 표준화함

· (보존포맷 선정기준 공공표준 제정) 전자기록 보존포맷 선정을 위한 공통기준과 문서 유형(텍스트, 스프레드시트, 프레젠테이션)에 대한 고유기준을 제시함

· (전자기록 유형별 고유기준의 다양화) 현재 보존포맷 선정체계는 문서 유형에 국한되어 있으며, 다양한 전자기록에 대해 보존포맷을 확대하고 다양화하는 것이 필요함

제2장 총괄연구개발과제의 최종 연구개발 내용 및 방법

2.1 연구개발과제의 내용

가. 데이터세트 유형 보존포맷 및 장기보존패키지의 현황 및 적합성 조사 및 분석

○ 데이터세트 보존포맷 및 이관·장기보존을 위한 방안에 대한 국내외 사례 조사

- NARA(미국), TNA(영국), SFA(스위스), NAA(호주) 및 PROV(호주 빅토리아주)의 데이터세트 보존포맷을 조사·분석함

- 기록관리 분야 이외에 산업, 문헌정보, 과학기술 및 컴퓨터 분야 등에서 RDB에 저장되어 있는 데이세트를 특수한 목적으로 이관 및 보존하는 사례를 조사·분석함

<표 1> 분야별 데이터세트 이관 및 보존 사례

분야	기관	관련 내용
문헌정보 분야	LOC (MARCXML)	MARC 데이터를 MARCXML로 변환하여 장기보존 (출처: https://www.loc.gov/standards/marcxml/)
	Europeana (EDM 통합 이관)	각 기관의 DB에서 추출한 메타데이터를 EDM(Europeana Data Model) 구조로 이관 (출처: https://pro.europeana.eu/page/metadata)
기업/산업 분야	SAP (SAP ILM)	데이터 아카이빙 및 시스템 폐기 관리 (출처: https://www.sap.com/products/technology-platform/information-lifecycle-management.html)
	IBM (InfoSphere Optim)	데이터 아카이빙 및 애플리케이션 은퇴 관리 (출처: https://www.ibm.com/products/infosphere-optim-archive)
과학기술 분야	CERN (CERN 오픈 데이터 포털)	입자 물리학 데이터를 공개 및 장기보존 (출처: https://opendata.cern.ch/)
	NASA (NASA DAAC)	지구 과학 데이터의 장기보존 및 접근성 유지 (출처: https://www.earthdata.nasa.gov/about/esdis/esco/standards-practices/nasa-earth-science-data-preservation-content-specification)
	GenBank(NCBI) (GenBank 개요)	유전자 서열 데이터의 수집 및 장기보존 (출처: https://www.ncbi.nlm.nih.gov/genbank/)
컴퓨터 분야	GitHub+Zenodo (Zenodo와 GitHub 연동)	소프트웨어 및 데이터세트의 장기보존 및 인용 가능성 제공 (출처: https://docs.github.com/ko/repositories/archiving-a-github-repository/referencing-and-citing-content)
	Dataverse Project (Dataverse 프로젝트)	연구데이터의 저장, 공유 및 장기보존 (출처: https://dataverse.org/)

- NARA(미국), TNA(영국), SFA(스위스), NAA(호주) 및 PROV(호주 빅토리아주)의 장기보존패키지 관련 조사 및 분석

<표 2> 장기보존패키지 관련 국제표준 및 해외 주요 사례

국가	기관	명칭
미국	LOC	BagIt
유럽	E-ARK	E-ARK CSIP/SIP/DIP/AIP
호주	PROV	VEO
국제표준	ISO 14721 OAIS 참조모형	정보패키지(Information Package)

○ 공공기관 행정정보시스템 DBMS 유형 분석 및 보존포맷 변환 적합성 검토

- 범정부EA 기반 공공부문 정보자원 현황 통계(~2024) 등을 통해 공공기관에서 보유 및 관리하고 있는 행정정보 시스템의 DBMS 종류를 파악하고, DBMS별 보존포맷 변환 적합성을 검토함

<표 3> 공공기관 DBMS 현황 통계

소프트웨어 유형	벤더명	수량	비율
DBMS	ORACLE	12,519	(63.52)
	MICROSOFT	3,160	(16.03)
	큐브리드	1,799	(9.13)
	티맥스데이터	1,621	(8.23)
	MARIADB	609	(3.09)
DBMS 합계		19,708	(100.00)

출처: 2025년 범정부EA 기반 공공부문 정보자원 현황 통계

나. (보존포맷 설계, 전북대) DBMS 데이터세트를 위한 오픈 포맷 활용 보존포맷 규격 설계

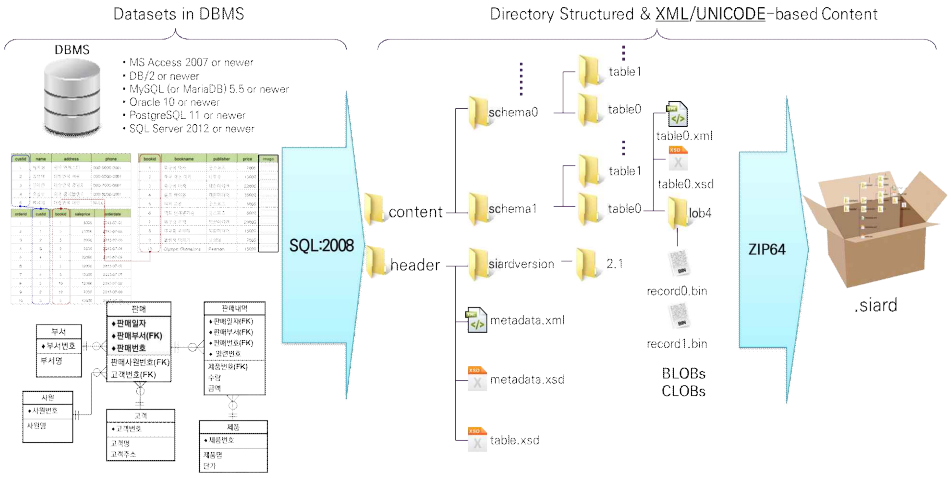
○ 데이터세트 보존포맷 SIARD(SIARD_KR 포함) 구조, 동작 프로세스 등 분석

- 2021년 8월 31일에 개정된 SIARD 2.2와 SIARD_KR에 의해 생성되는 .siard 파일포맷의 구조를 분석함

<표 4> SIARD 2.2 및 SIARD_KR 개요

국가(기관명)	표준명	출처
한국(NAK)	SIARD_KR	https://github.com/nakdataset/SIARD_KR
스위스(SFA)	SIARD 2.2	https://github.com/DILCISBoard/SIARD

- SFA와 E-ARK 프로젝트에서 제·개정된 SIARD 표준은 DBMS에 저장된 데이터세트를 'SQL:2008'에서 정의된 SQL 쿼리문을 기준으로 추출하여 'XML' 구조와 'UNICODE' 인코딩 방식으로 변환 및 저장함. 이때 메타데이터 정보는 header 폴더에, 실제 데이터 정보는 content 폴더에 저장되며, 최종적으로는 header 폴더와 content 폴더를 'ZIP64' 방식으로 패키징함. 이 과정은 아래 <그림 3>에서 확인할 수 있음



<그림 3> SIARD 파일 생성과정

출처: eCH-0165 SIARD Format Specification ver2.1

❍ 데이터세트의 이관·보존을 위한 새로운 보존포맷 형태 도출 및 규격 작성

- (구조 정의) 다양한 DBMS에 적용 가능한 보존포맷 규격을 설계하기 위해, 각 DBMS에서 정의·운영되는 구성요소를 식별하고, 이를 구조적으로 정리함. 이를 위해 DDL(Data Definition Language)을 통해 정의되는 스키마(Schema) 정보, DML(Data Manipulation Language)을 통해 생성·조작되는 실제 데이터, DCL(Data Control Language)을 통해 설정·관리되는 사용자 권한 및 접근제어 정보 등을 체계적으로 분석하고, 보존포맷에 포함될 수 있도록 정형화함

· DML과 DDL에 의해서 생성되는 데이터는 SIARD를 참고하여 구성하는 것이 합리적임. 재현을 위해서 다른 구조의 XML 파일로 변환할 때, XSLT와 같은 표준화된 방식으로 변환과정을 명세할 수 있는 방법을 도출함

· DCL에 의해서 생성되는 데이터는 아래와 같이 정리할 수 있으며, DBMS에 적용된 명령어 자체를 트랜잭션 로그나 감사(Audit)로그 등에서 추출하거나, 시스템 카탈로그 뷰나 정보 스키마(INFORMATION_SCHEMA) 등을 통해서 구조화된 메타데이터 형태로 제공받을 수 있음

<표 5> DCL 생성 데이터 유형

항목	세부항목	내용
주체 (Subject)	사용자 (User)	권한을 부여받거나 행사하는 개체 (예. CREATE USER)
	역할 (Role)	권한을 부여받거나 행사하는 개체 (예. CREATE ROLE)
권한 (Privileges)	객체 권한 (Object Privilege)	주체가 행사할 수 있는 DB 작업 권한 (예. GRANT SELECT ON employees TO user1)
	시스템 권한 (System Privilege)	DB 생성, 사용자 생성 등 시스템 차원의 권한 (예. GRANT CREATE SESSION TO user1)
	권한 위임 (Grant Option)	부여받은 권한을 제3자에게 다시 부여할 수 있는 권한 (예. GRANT SELECT ON emp TO user2 WITH GRANT OPTION)
권한 제어 명령 (Control Commands)	-	권한을 부여하거나 철회하는 SQL 명령어 (예. GRANT, REVOKE)
보안 정책 (Security Policy)	접근 제어 정책 (Access Control Policy)	행 수준 접근 제어 또는 조건부 제한 설정 (예. Oracle VPD, PostgreSQL RLS)
	감사 정책 (Audit Policies)	특정 행위 또는 객체에 대한 접근 기록 설정 (예. AUDIT SELECT TABLE BY hr_user)

- (오픈포맷 선정) 데이터 패키징 방식(Base64 인코딩, 디렉토리 구조화, 압축 방식 등)을 검토하고, 실제 데이터, 스키마 구조, 접근제어 정보를 저장할 수 있는 오픈 포맷(XML, XSLT, CSV, JSON 등)의 적합성을 분석하여 최적의 방식을 도출함

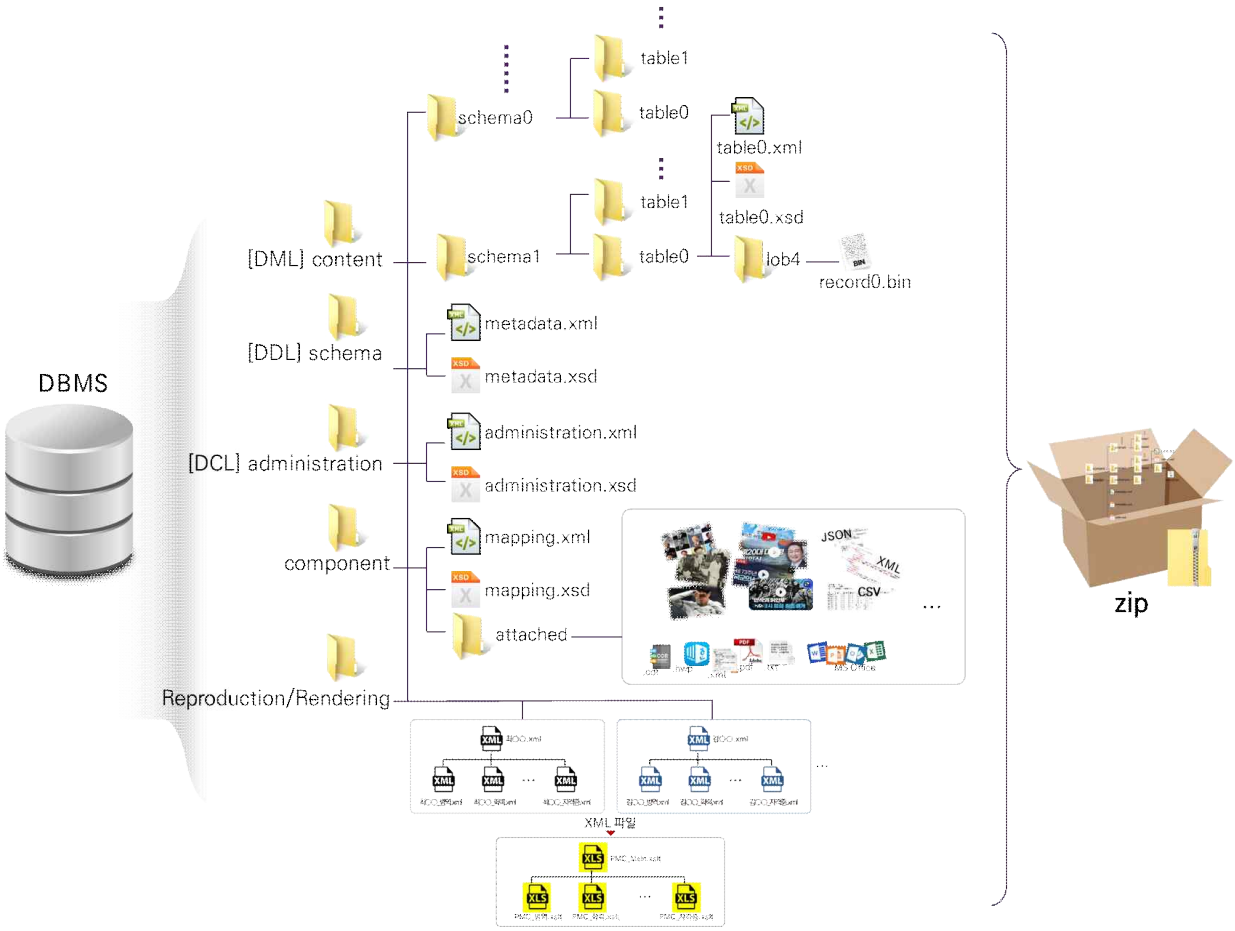
- (패키징 방안 도출) BLOB 및 CLOB과 같은 바이너리 대용량 데이터와 사용자 정의 함수(User Defined Function), 저장 프로시저(Stored Procedure)와 같은 함수형 루틴의 저장 및 보존 방안도 함께 도출함

- (첨부파일 저장) 데이터베이스에서 파일 자체를 저장하지 않고, 파일의 경로 정보(URL, 파일 경로 등)만을 참조하는 방식으로 운영되는 경우가 많음. 이러한 구조에서는 데이터베이스 내 정보만으로는 실제 파일을 확보하거나 재현하기 어려우므로, 외부 파일까지 포함할 수 있는 보존포맷 구성 방안을 함께 모색할 필요가 있음. 이를 통해 데이터베이스 외부의 파일시스템에 저장된 첨부파일까지 연계하여 통합적으로 관리할 수 있는 장기보존 구조를 설계하고자 함

- (서식 재현용 보존포맷 규격 정의) 데이터세트를 서식 형태까지 재현하기 위해서는, 관계형 데이터베이스의 원본 구조를 그대로 유지하기보다는 XML/XSLT 기반의 재현 가능한 형태로 변환하는 방식이 필요함. 이때 데이터의 진본성을 확보하기 위해 변환(마이그레이션) 절차를 표준화하고, 해당 절차를 명확한 기술 명세(Specification)로 정의하는 방안을 함께 마련함

- (규격 작성) 생산 형태(DB) 그대로 보존하는 방식, 서식 형태로 보존하는 방식 등을 구분하여 데이터세트용 보존포맷 규격을 작성함

- 보존포맷의 전체 규격이 작성되면 <그림 4>와 같은 구조와 생성 프로세스(안)로 생성될 수 있음

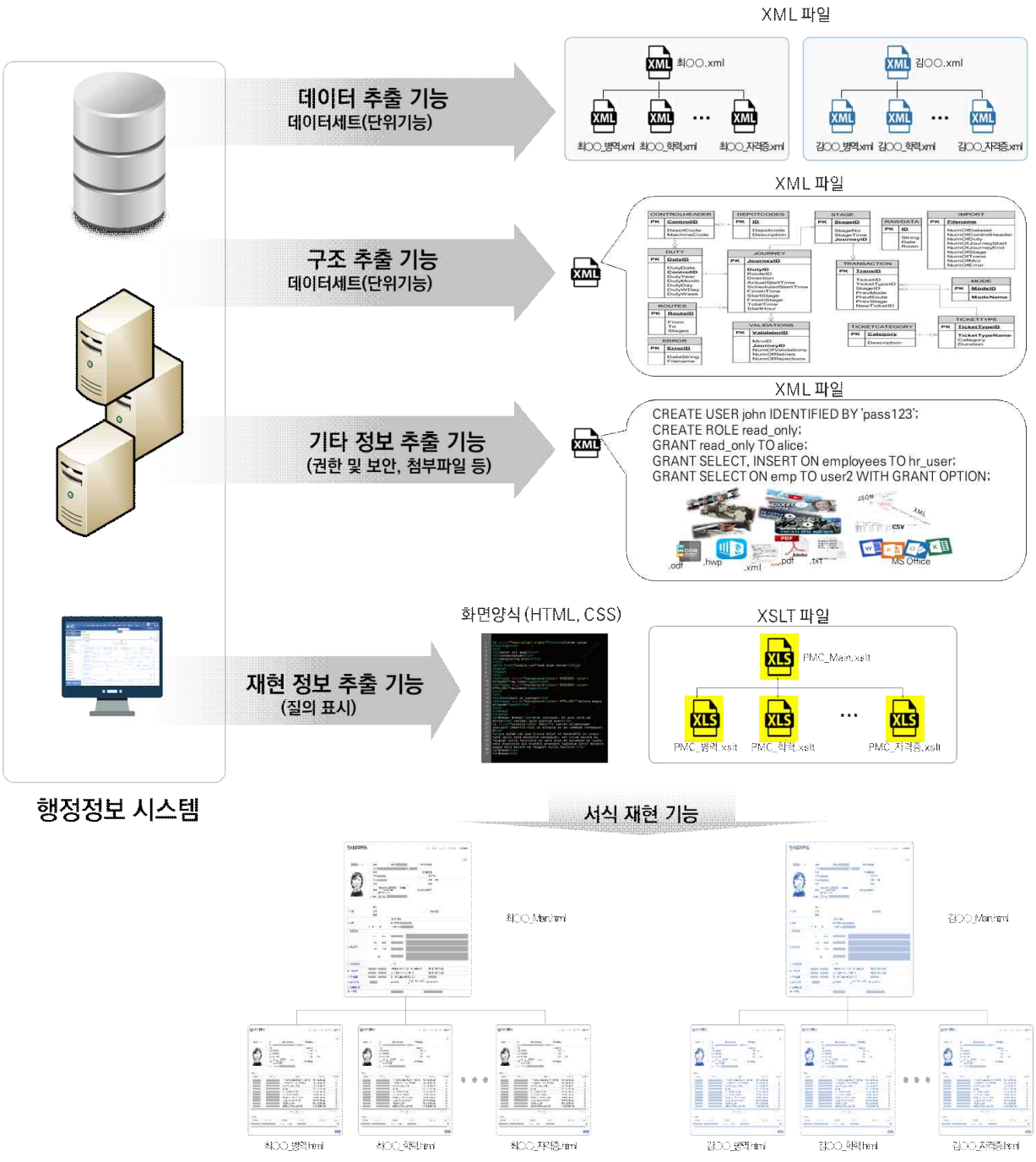


<그림 4> 데이터세트 유형 보존포맷 구조 및 생성 프로세스(안)

다. 도출된 보존포맷의 변환·재현 테스트베드 구축 및 실데이터 기반 검증

○ 실데이터를 활용한 보존포맷 변환 가능여부 및 변환 시 유의사항 도출

- 보존포맷의 변환·재현을 위해 아래처럼 데이터 추출 기능, 구조 추출 기능, 기타 정보 추출 기능(권한 및 보안, 첨부파일 등)을 구현하여 테스트베드를 구축함



<그림 5> 보존포맷의 변환·재현을 위한 테스트베드 기능

- 다양한 DBMS에서 운영 중인 데이터세트를 대상으로, 보존포맷 규격에 따른 변환 가능 여부를 실증적으로 검토하고, 데이터 유형별 변환 결과를 분석함

- DDL, DML, DCL, 첨부파일, 대용량 바이너리 데이터(BLOB/CLOB), 함수형 루틴 등 각 요소에 대한 변환 적용 방식과 한계점 및 예외 사례를 식별함

- DBMS 간 호환성, 인코딩 방식, 데이터 정합성 유지 여부 등 변환 시 발생 가능한 오류 및 유의사항을 도출하고, 변환 자동화 도구 적용 가능성과 한계도 함께 검토함

○ 실데이터를 활용한 보존포맷에 따른 재현방안 도출 및 검증

- 데이터 추출 기능으로 생성된 XML 파일과 재현 정보 추출 기능을 통해 생성된 XSLT 스타일시트를 활용하여, 실제 화면 양식을 재현할 수 있는지를 실데이터 기반으로 검증함. 이를 통해 XML 기반 데이터와 XSLT 변환 로직 간의 연계 완성도 및 시각적 재현 정확성을 평가함

- 보존포맷으로 변환된 데이터세트를 기반으로, DBMS 환경에서의 구조적 복원과 XML/XSLT 기반의 서식 재현 방식을 각각 수행하고, 그 절차 및 결과를 비교 검토함

- 재현 과정에서 필요한 구성요소(데이터, 스키마, 권한 정보 등)의 최소 요건을 도출하고, 데이터 일관성, 참조 무결성, 형식 재현의 정확도 등을 중심으로 품질을 평가함

- 다양한 보존포맷 형태(생산형, 서식형 등)에 따른 재현 전략을 분석하고, 향후 실무 적용을 위한 기술적 요건 및 절차를 정립함

라. 데이터세트 유형에 적합한 NEO3의 구성요소와 구조를 정의한 장기보존패키지 규격 설계

○ 데이터세트 유형에 대한 NEO3의 구성요소 도출 및 정의

- NEO3 장기보존패키지의 구성요소인 원문, 보존포맷, 장기보존 메타데이터의 구조와 내용을 데이터세트 유형에 적합한 방식으로 구체적으로 정의함. 이를 통해 데이터세트의 특성과 보존 요구에 부합하는 장기보존패키지 구성을 마련함

· 데이터세트의 원문은 일반적인 파일 형태의 독립된 정보 객체가 아니라, 현재 운영 중인 관계형 데이터베이스 내에 저장된 실시간 데이터임. 이 데이터는 대용량일 뿐만 아니라, 국내 데이터세트의 기록관리 단위로 세분화하여 덤프(Dump)하는 것이 기술적으로 복잡하고 비효율적임. 전체 데이터베이스를 그대로 덤프하는 방식은 기록관리 목적에 부합하지 않으며, 덤프 파일 역시 DBMS의 버전이나 구조가 변경될 경우 복원이 어려운 사례가 다수 발생함. 이에 따라 데이터세트 유형의 기록물에서 원문을 반드시 정의해야 하는지 여부에 대한 검토가 필요하며, 정의가 필요한 경우에는 현실적인 구성 방안을 함께 도출해야 함

· 호주 PROV의 「PROS 19/05 S3: Long Term Sustainable Formats Specification」에 따르면, 기록의 분량이 방대하거나 변환과정에서 콘텐츠 손실 가능성이 거의 없는 경우에는 원문의 보존을 필수로 요구하지 않음. 이러한 기준은 원문을 반드시 장기보존패키지에 포함해야 한다는 절대적인 요구가 아님을 시사하며, 데이터베이스 기반의 대용량 데이터세트처럼 원문 추출이 비효율적이거나 복원이 어렵고 실익이 적은 경우, 원문 포함을 선택적으로 판단할 수 있는 방향성을 제시함

> ○ PROS 19/05 S3: Long Term Sustainable Formats Specification
> - 이관을 목적으로 기록물의 콘텐츠가 장기보존포맷으로 변환된 경우, PROV와 별도의 합의가 없는 한 미변환된 원본 포맷의 사본 또한 VEO에 포함되어야 한다.
> - 하지만 방대한 양의 기록이거나 콘텐츠 손실 가능성이 거의 없는 변환 과정이라면 원본 포맷의 사본은 요구되지 않는다.

· 보존포맷은 본 연구과제의 세부과제인 '보존포맷 설계'에서 도출된 신규 보존포맷 규격을 기준으로 설정할 수 있음

· NEO3를 정의하고 있는 「공공표준 NAK 31-2」의 장기보존 메타데이터는 「공공표준 NAK 8」이 정의하고 있는 기록관리 메타데이터를 그대로 설정함. 이때, 기록관리 메타데이터는 공

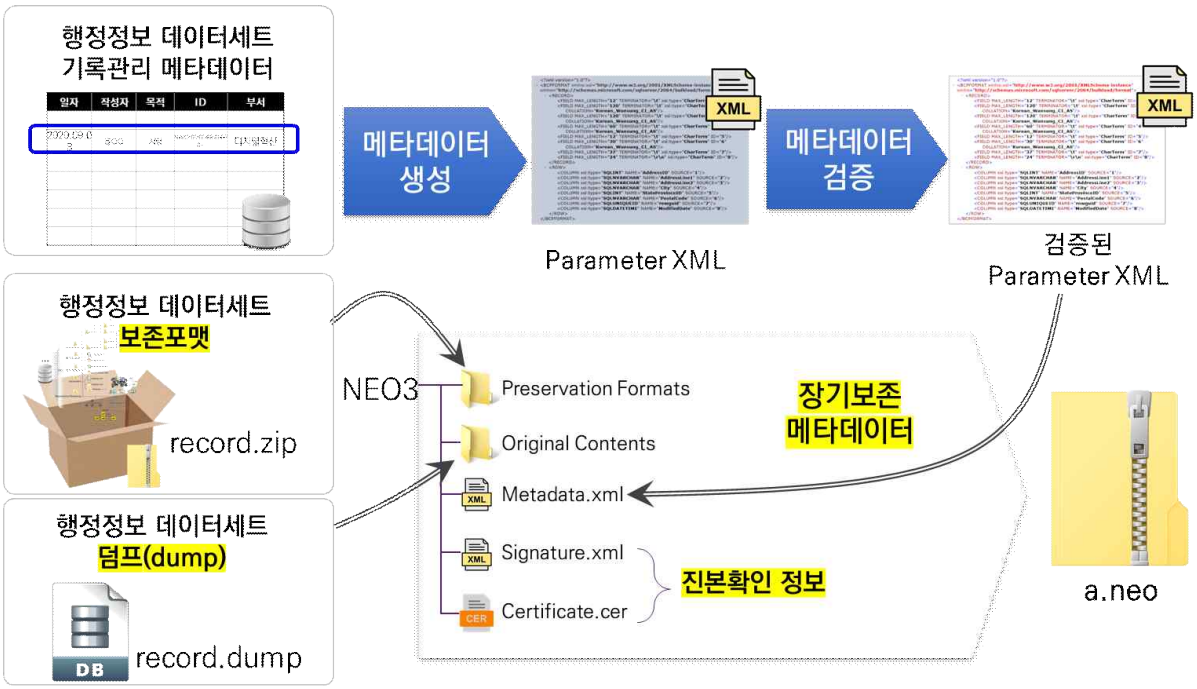
문서에 특화된 메타데이터로 공문서 이외에 다른 기록물에는 적합하지 않은 항목들이 많고, 데이터세트에 반드시 필요한 내용이 없으므로 데이터세트에 그대로 적용하는 것이 불가능함. 따라서 데이터세트 기록관리를 위한 메타데이터를 새롭게 정의해야 함. 이때, 「공공표준 NAK 35」에 정의되어 있는 데이터세트 관리기준표 또는 학계에서 도출된 아래의 연구결과를 활용, 참고하여 정의할 수 있음

- 국가기록원 공공표준 NAK 35:2020(v1.0(2020). 행정정보 데이터세트 기록관리 기준 - 관리기준표 작성 및 이관규격
- 이정은, 김지혜, 왕호성, 양동민. (2022). 행정정보 데이터세트의 관리기준표 개선방안 연구. 한국기록관리학회지, 22(1), 177-200.
- 김지혜. "행정정보 데이터세트 기록관리를 위한 메타데이터 요소 연구." 국내석사학위논문 전북대학교 일반대학원, 2023. 전북특별자치도

○ 데이터세트 유형에 적합한 NEO3 표준 규격 작성

- NEO3는 다양한 전자기록물 유형을 수용할 수 있도록 설계됨. 다만, 해당 구성요소들이 NEO3의 구조에 맞게 폴더(Root, Original Contents, Preservation Formats, Components)에 잘 배치되고, 배치된 정보가 NEO3의 PackagingInformation.xml, Metadata.xml, Signature.xml 등 패키징 정보, 기록관리 메타데이터, 진본확인 정보에 정확하게 명시함

- 다음 <그림 6>은 데이터세트 유형 NEO3 장기보존패키지의 생성과정을 도식화한 것임. 새롭게 정의된 보존포맷과 데이터세트 메타데이터를 활용하여, 데이터세트 기록물이 NEO3로 패키징되는 구체적인 프로세스를 명세하여 NEO3 표준 규격을 작성함

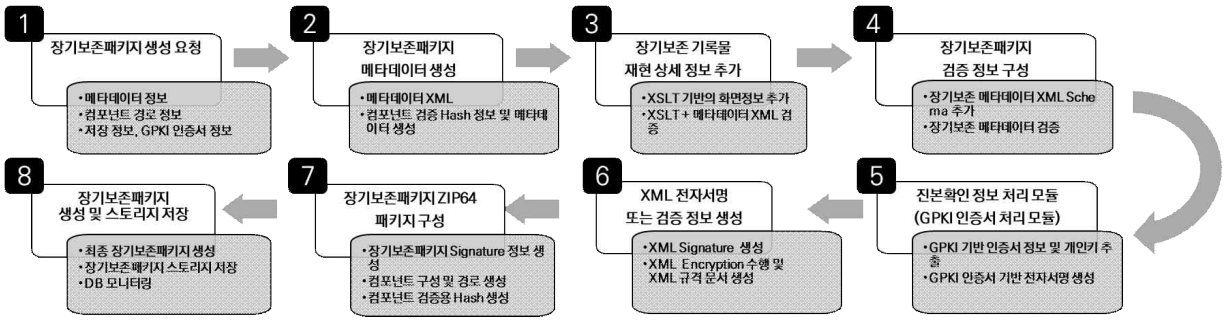


<그림 6> 데이터세트 유형 NEO3-장기보존패키지 생성과정

마. NEO3 장기보존패키지의 변환 테스트베드 구축과 실데이터 기반 검증

○ NEO3 구조 기반 패키징 기능 시범 구현

- 시범 개발된 기능은 새롭게 정의된 보존포맷, 데이터세트 유형 메타데이터 요소, 진본확인 정보 등을 NEO3 구조(폴더 구성 및 XML 기반 내부 파일 구조)에 맞게 <그림 7>의 절차에 맞게 패키징할 수 있도록 구현함



<그림 7> NEO3-장기보존패키지 생성을 위한 세부 기능의 단계적 수행 절차

○ 실데이터 기반 NEO3 패키지 생성 검증 및 기술적 완성도 평가

- 다양한 유형의 데이터세트를 실제로 입력값으로 활용하여, NEO3 규격에 따른 생성 절차와 구성요소 정합성을 검증하고, 패키지 내부 정보의 자동화 수준, 구조의 일관성, 오류 발생 사례 등을 중심으로 기술적 타당성과 완성도를 평가함

2.2 연구개발과제의 방법

가. (문헌사례 분석) 데이터세트 유형 보존포맷 및 장기보존패키지의 현황 및 적합성 조사 및 분석

○ 연구전략

- 데이터세트 기록관리와 장기보존에 관한 국내외 보존포맷 및 패키지 사례를 체계적으로 수집하고, 기록관리·문헌정보·컴퓨터공학 등 다양한 분야를 포함하여 분석함으로써 데이터세트 유형에 적합한 보존방식과 구조를 비교·평가함

○ 연구방법론

- 주요 국가기관(NARA, PROV, E-ARK 등)의 사례, 표준, 기술 등 문헌 및 사례를 분석함
- 보존포맷 및 장기보존패키지의 구성요소, 기술 구조, 적용 방식 등을 기준으로 비교·검토함
- 각 보존포맷 및 장기보존패키지의 장단점과 적용 가능성을 평가하여 본 과제의 보존체계 설계에 활용할 기초자료로 정리함

나. (보존포맷 설계) DBMS 기반 데이터세트를 위한 오픈 포맷 활용 보존포맷 규격 설계

○ 연구전략

- 다양한 DBMS에 공통으로 적용할 수 있는 보존포맷을 설계하기 위해, DDL/DML/DCL 구성요소를 기준으로 데이터를 식별하고, 이를 XML, JSON, CSV 등 오픈 포맷 기반으로 구조화

○ 연구방법론

- DB 구성요소(스키마, 데이터, 접근 권한 등)에 대한 구조적 요구사항을 분석하고, 이를 오픈 포맷에 매핑할 수 있는 방법을 도출하며, SIARD 등 선행 보존포맷의 구조를 벤치마킹하여 BLOB/CLOB, Function/Stored Procedure 등 특수 객체의 처리 방안을 포함함
- 다양한 보존포맷의 장단점을 기술적 관점에서 평가하고, 본 과제에서 활용할 신규 보존포맷 규격을 설계함

다. (보존포맷 검증) 보존포맷의 변환·재현 테스트베드 구축 및 실데이터 기반 검증

○ 연구전략

- 설계된 보존포맷이 실제 데이터세트에 적용 가능한지 실증적으로 확인하기 위해 테스트베드를 구축하고, 다양한 DBMS의 실데이터를 활용하여 변환 및 재현 가능성을 검증함

○ 연구방법론

- 테스트베드를 구축하여 실데이터를 보존포맷으로 변환하고, 구조·데이터·권한 등 구성요소의 손실 여부를 점검함
- 변환 도구 적용 시 예외 상황 및 오류 사례를 수집·분석하고, DBMS별 유의사항을 도출함

라. (NEO3 설계) 데이터세트 유형에 적합한 NEO3의 구성요소와 구조를 정의한 장기보존패키지 규격 설계

○ 연구전략

- NEO3 구조를 데이터세트 유형에 최적화하기 위해, 기존 NAK 31-2 규격을 검토하고, 보존포맷과 데이터세트 메타데이터를 반영한 새로운 패키지 구성요소와 구조를 설계함

○ 연구방법론

- 기존 표준의 구성요소(원문, 보존포맷, 메타데이터 등)의 정의 및 적용성을 분석하고, 한계점을 보완하여 데이터세트에 적합한 메타데이터를 재정의하고, 폴더와 XML 데이터를 구체화함
- 데이터세트에 특화된 메타데이터 요소를 도출하고, 국내외 표준 및 학술연구 결과를 반영하여 패키지 규격을 작성함

마. (NEO3 검증) NEO3 장기보존패키지의 변환 테스트베드 구축과 실데이터 기반 검증

○ 연구전략

- 설계된 NEO3 규격의 실효성을 검증하기 위해, 실데이터를 기반으로 패키지 생성 기능을 구현하고, 다양한 유형의 데이터세트에 적용하여 재현 가능성과 완성도를 검토함

○ 연구방법론

- NEO3 패키지를 자동 생성하는 기능을 시범 개발하고, 실제 데이터세트를 입력하여 구성요소별 패키징 결과를 확인하며, 재현 과정에서 XML/XSLT 기반의 서식 출력, 원형 데이터 복원 등 다양한 재현 방식을 검증함
- 구성요소 간 연계성, 진본성 확보, 메타데이터 정합성 여부 등을 기준으로 실효성을 평가하고 개선점을 도출함

제3장 총괄연구개발과제의 최종 연구개발 결과

3.1 데이터세트 보존포맷 및 장기보존패키지의 현황 및 적합성 조사 및 분석

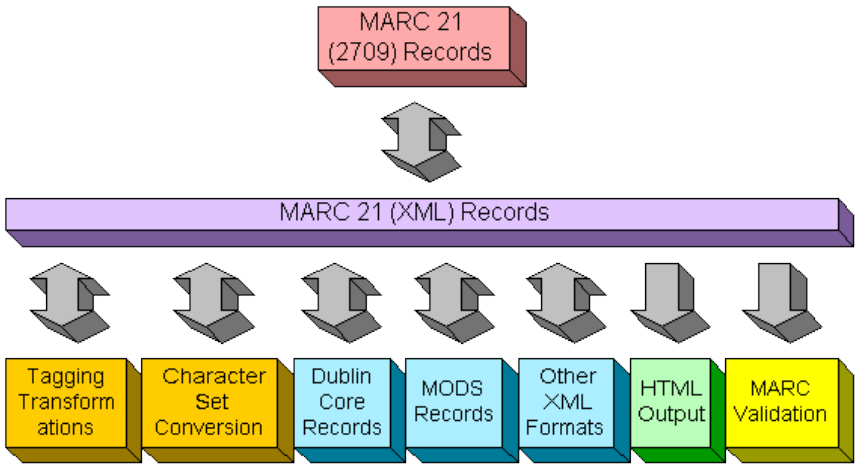
가. 국내외 아카이브 기관 데이터세트 보존포맷 및 장기보존패키지의 현황 조사 및 분석

○ **(기록관리)** 미국 의회도서관(LOC)의 MARCXML

- 개요

· MARCXML은 미국 의회도서관(LOC)의 Network Development and MARC Standards Office에서 1990년대 중반에 개발한 프레임워크임

· 기존 MARC(ISO 2709) 데이터는 XML 환경에서 활용할 수 있도록 설계되었고, 디지털 환경에서 유연한 메타데이터 기술과 상호운용성을 확보하는 데 목적을 두고 있음(<그림 8> 참조)



<그림 8> MARCXML의 구조

· MARC 21 완전 재현: MARC 레코드를 XML 형식으로 손실 없이 변환 가능함

· MARC 데이터를 Dublin Core, MODS 등 다양한 메타데이터 형식으로 변환할 수 있는 기반을 제공함

· 메타데이터에는 해시값을 활용하여 무결성을 검증할 수 있는 기능을 포함하고 있음

- 평가 및 분석

· 다양한 시스템에서 활용 가능: MARCXML은 책이나 기록을 설명하는 서지정보 중심 메타데이터 형식임. XML 기반이라 도서관 시스템, 기록관리 시스템, 아카이빙 시스템 등 다양한 환경에서 손쉽게 활용 가능함. 특히 데이터의 구조화된 표현을 통해 시스템 간 호환성과 상호운용성을 보장할 수 있음. 다만, 데이터베이스의 실제 데이터를 보관하기보다는 자료를 설명하는 정보(메타데이터)를 보존하는 데 더 적합함

· 개방형 표준 포맷: MARCXML은 미국 의회도서관이 만든 공개 XML 표준으로, 누구나 자유롭게 사용 가능함. 또한 MODS, Dublin Core, RDF 같은 다른 메타데이터 표준으로 변환할 수 있어 국제표준 기반의 상호운용성 확보에 유리함

○ **(기록관리)** Europeana(유로피아나)의 EDM(Europeana Data Model)

- 개요

· EDM은 박물관, 도서관, 기록관 등 다양한 문화유산 기관의 메타데이터를 통합하고 연결할 수 있도록 만든 모델임. LIDO, EAD, METS 등 여러 표준을 포괄하는 개방형 시맨틱웹 기반 프레임워크임

· 기존에 사용하던 ESE(Europeana Semantic Elements)가 표현할 수 있는 정보가 제한적이었던 문제를 개선해, 유연하게 데이터를 표현할 수 있도록 설계됨

· EDM은 데이터 생산자가 제공한 정보, 실제 객체(원본 자료), 그리고 웹에서 접근할 수 있는 디지털 객체를 명확히 구분할 수 있게 해줌. 이를 통해 유로피아나는 제공된 데이터에 Linked Data를 적용해 리소스 간 새로운 관계를 만들어 정보를 확장 가능함

- 설계 원칙

· 개방성: 다양한 형태의 데이터를 받아들일 수 있는 환경을 제공함

· 확장성: 필요할 때 기능과 표현 범위를 유연하게 넓힐 수 있음

· 표준 재사용: 이미 사용되는 표준 모델을 최대한 활용해 상호운용성을 높임

- 메타데이터 기술 방식

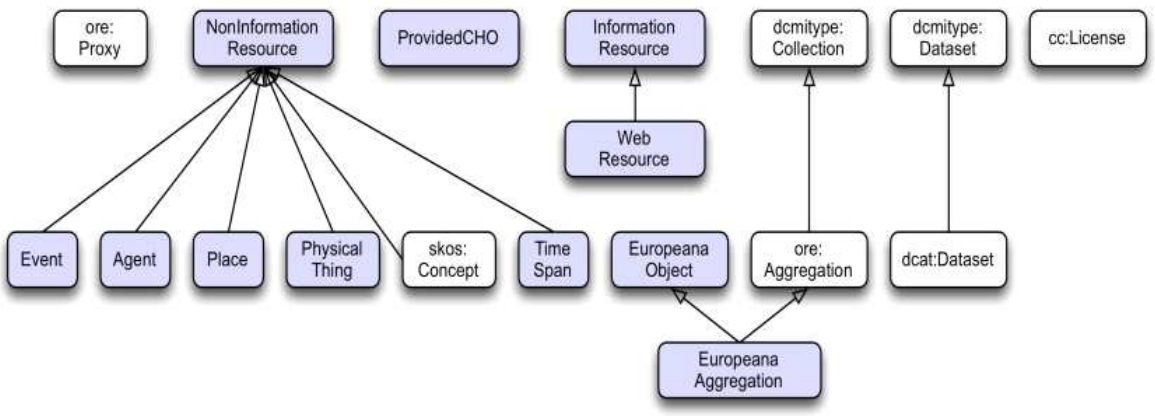
· EDM은 관련 자료를 하나로 묶어서 설명할 수 있게 하는 방식으로 사용함. 예를 들어, 박물관에서 유물을 기록한다고 하면, 이 유물 자체에 대한 정보뿐 아니라 사진, 설명문서, 영상자료, 관련 논문 등과 같은 여러 리소스를 하나로 묶어서 관리할 수 있음. 이를 '집합(Aggregation)'이라고 칭함

· EDM은 두 가지 방법으로 정보를 표현할 수 있음. 객체 중심(Object-centric)은 기록하고자 하는 대상 자체에 초점을 맞춰 기술함. 그리고 이벤트 중심(Event-centric)은 기록하고자 하는 대상의 생성된 과정, 사건 등과 같은 맥락 정보를 함께 기록함

- EDM 구조

· EDM 클래스 계층 구조는 <그림 9>와 같음.

· EDM 고유의 클래스는 연한 보라색 사각형으로 표시되어 있으며, 다른 스키마와 공통으로 사용되는 클래스는 흰색 사각형으로 표시됨



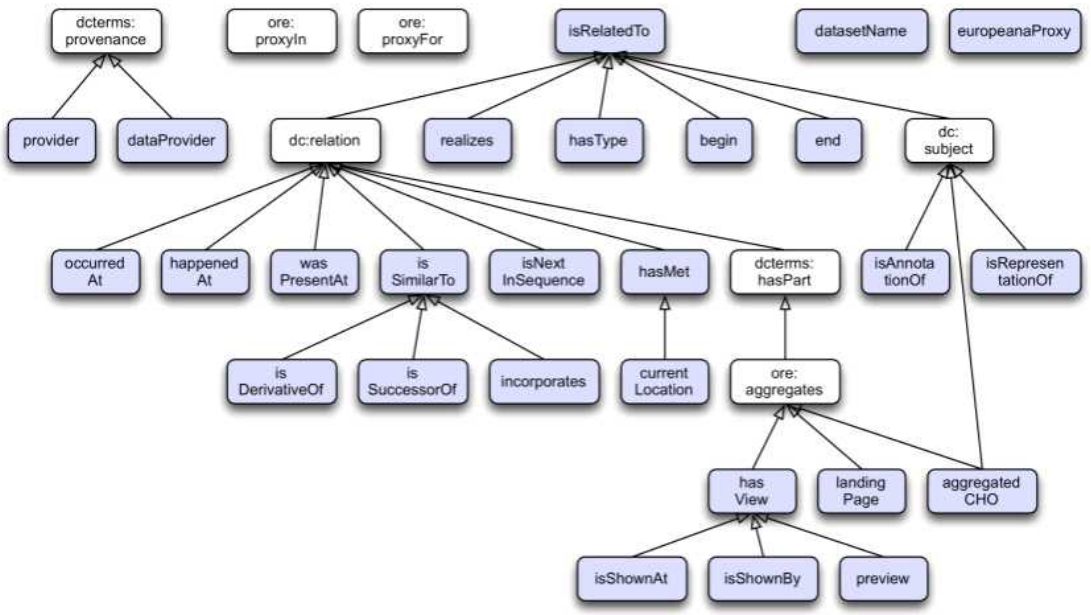
<그림 9> EDM 클래스 계층 구조

· EDM의 고유 클래스는 <표 6>과 같음

<표 6> EDM의 고유 클래스

클래스	정의
Non-Information Resource	Information Resource 외 모든 리소스(예: 사람, 장소, 실물 객체 등)
Provided Cultural Heritage Object	유로피아나가 수집하는 문화유산 객체
Information Resource	본질적인 객체의 특성(예: 텍스트 문서, 디지털 객체 등)을 나타내는 리소스임. 이 리소스는 URL로 연결될 수 있음
Web Resource	웹상에서 URL을 통해 접근이 가능한 정보 리소스
Event	일정한 시간과 공간 안에서 발생한 의미 있는 변화나 활동을 포괄하는 개념으로, 문화유산의 맥락 정보를 기술하는 데 중요한 역할을 함
Agent	데이터를 생산·소유·제공·관리하는 개인이나 기관
Place	시간이나 사물과는 분리된 물리적으로 정의된 위치나 장소를 의미함
Physical Thing	사람을 제외한 물리적인 실체를 의미함. 유로피아나에서 물리적인 실체로 간주하는 문화유산 객체와 유로피아나가 문화유산 객체를 설명할 때 언급하는 모든 물리적인 대상을 포함함
Time Span	하나의 사건이나 활동이 발생한 기간(예: 역사적 시대 등)
Europeana Object	유로피아나의 활동 결과로 생성된 모든 객체
Europeana Aggregation	하나의 문화유산 객체를 기술하기 위한 데이터들의 집합

· EDM의 고유 속성들은 연한 보라색 사각형으로 표시되어 있으며, 다른 스키마와 공통으로 사용되는 속성들은 흰색 사각형으로 표시됨(<그림 10> 참조)



<그림 10> EDM 주요 속성

· EDM 고유 속성 중 일부는 <표 7>과 같음

<표 7> EDM의 고유 속성(일부)

클래스	정의
isRelatedTo	EDM에서 가장 일반적인 맥락 정보임
datasetName	유로피아나에서 해당 데이터세트에 부여한 식별자
europeanaProxy	프록시(proxy)가 외부 프록시가 아니라 유로피아나 프록시임을 표시하는 속성임. 내부에서만 사용됨
provider	유로피아나에 데이터를 제공하는 기관의 이름 또는 식별자
dataProvider	유로피아나에 데이터를 제공하는 기관의 이름 또는 식별자
realizes	물리적 객체와 정보 자원 간의 관계를 설명함
hasType	리소스를 MIME과 같은 적절한 유형 체계나, 특정 분야의 객체 범주를 분류하는 시소러스 내 개념들과 연결함
begin	특정기간의 시작 날짜
end	특정기간의 마지막 날짜

- 평가 및 분석

· EDM은 RDF 기반의 개방형 포맷을 사용하여 특정 데이터베이스(RDBMS, NoSQL 등)에 종속되지 않음. 다양한 시스템에서 생성된 메타데이터를 통합·매핑하여 하나의 표준화된 데이터 모델로 표현 가능함. 이를 통해 이기종 시스템 간 상호운용성 확보에 유리함

· EDM은 디지털 객체를 직접 저장하지 않고, WebResource 클래스를 통해 외부 저장소에 있는 이미지·오디오·비디오 등의 대용량 파일을 URI로 연결함. 이를 통해 메타데이터와 실제 파일을 분리하면서도 참조 관계를 유지할 수 있어 보존 및 접근의 유연성을 강화함

· EDM은 국제표준 RDF/XML 포맷을 기반으로 하며, 스키마 정의와 문서가 모두 공개되어 있음. SKOS, Dublin Core(DC), OWL 등 국제적으로 널리 사용되는 표준을 채택하고 있으며, MODS·METS 등 다른 메타데이터 표준으로의 상호 변환도 지원함. 이러한 오픈 포맷과 국제표준 중심의 설계는 장기적인 호환성과 확장성을 보장하고, 보존포맷 설계 시 URI를 통해 외부 저장소의 대규모 파일을 연결하는 방식을 손쉽게 구현할 수 있게 함

○ **(기록관리)** University of Minho(민호 대학교)의 Jose Carlos Ramalho 연구팀

- 개요

· 관계형 데이터베이스는 테이블, 열, 행 간의 복잡한 구조와 데이터 간의 다양한 연계 관계를 지니고 있어, 기존 디지털 보존방식만으로는 완전한 장기보존을 보장하기 어려움

· 데이터베이스 내 정보는 다양한 유형과 구조를 포함하고 있어 시간의 흐름에 따라 일관성을 유지하기 어려우며, 각 데이터베이스 관리 시스템(DBMS)마다 저장 포맷과 질의 언어가 상이하여 상호운용성과 변환과정에서 기술적 어려움이 발생함

· 이러한 문제를 해결하기 위해 스위스 연방 기록원과 E-ARK 프로젝트는 데이터베이스의 구조와 의미를 손상시키지 않으면서 장기적으로 보존할 수 있는 SIARD 2.0 포맷을 개발함

· SIARD 2.0는 XML, Unicode, URI와 같은 국제표준 기술을 기반으로 하며, 업그레이드된 SQL 표준(ISO 9075:2008)을 채택하여 다양한 데이터베이스 요소(배열, 사용자 정의 타입 등)

를 안정적으로 보존할 수 있음

· 데이터베이스 보존 과정에서 발생할 수 있는 수작업의 한계와 오류 가능성을 최소화하기 위해 DBPTK(Database Preservation Toolkit)가 활용됨

· DBPTK는 데이터베이스로부터 데이터와 메타데이터를 자동으로 추출하고 SIARD 2 형식으로 변환하며, 필요 시 다른 DBMS로 복원할 수 있는 기능을 제공함

· 이 도구는 파일럿 테스트와 라운드트립 테스트를 통해 변환의 정확성과 보존 품질이 검증되어, 실제 기록관 환경에서 대규모·복잡한 데이터베이스를 안정적이고 신뢰성 있게 장기 보존할 수 있도록 지원함

- SIARD 2.0의 시스템 보존 절차

· 관계형 데이터베이스의 중요한 속성을 장기적으로 보존하기 위해, 스위스 연방 기록원과 E-ARK 프로젝트는 협력하여 SIARD 2.0 포맷을 개발함

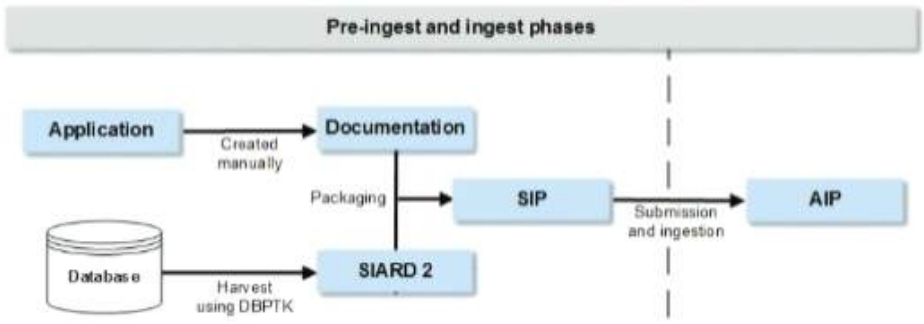
· SIARD 2.0는 안정적이고 보편적인 기술을 기반으로 구성되어 있음. 데이터 구조와 내용을 표현하기 위하여 XML을 활용하고, 다국어 문자 표현을 위해 국제표준인 Unicode를 사용하며, 자원 식별을 위한 통일된 방식으로 URI를 채택함

· 또한, SIARD 2.0는 초기 버전보다 업그레이드된 SQL 표준(ISO 9075:2008)을 채택하였으며, 배열(array), 사용자 정의 타입(user-defined types) 등 더 다양한 데이터베이스 구성요소를 지원함

· 데이터베이스 보존은 '사전 수집(pre-ingest)'과 '수집(ingest)'의 두 단계로 나뉨(<그림 11> 참조)

· 사전 수집 단계에서는 데이터베이스가 사용된 시스템의 기능, 구조, 활용 방식을 설명하는 문서를 미리 준비해 보존 맥락을 확보함. 그리고 DBPTK라는 도구를 사용해 기존 데이터베이스를 장기보존 포맷인 SIARD 2.0 파일로 변환함

· 수집 단계에서는 변환된 SIARD 2.0 파일과 설명문서를 묶어 하나의 패키지(SIP)로 만듦. 이 패키지를 아카이브에 제출하면, 보존소가 데이터를 검증한 뒤 AIP(Archival Information Package)로 변환해 안전하게 장기 보존함



<그림 11> SIARD 2.0 데이터베이스 및 시스템 설명문서의 패키징 및 제출 절차

- SIARD 2의 이용자 접근 절차

· 아카이브는 설명문서와 SIARD 2.0 형식의 데이터베이스를 포함하는 DIP(Dissemination Information Package)를 생성함

· 이용자는 이 DIP를 받아, DBPTK를 이용해 SIARD 2.0 데이터베이스를 다시 DBMS에 불러올 수 있음

· OLAP(온라인 분석 처리) 또는 데이터 마이닝 등의 지식 발견 기술을 사용하는 시스템이 있

다면, 이를 통해 보존된 데이터를 분석할 수 있음

- DBPTK(Database Preservation Toolkit)

· 관계형 데이터베이스를 보존하려면, 데이터와 메타데이터를 추출하고 이를 장기보존포맷으로 변환하는 과정이 필요함. 하지만 이 작업을 수작업으로 진행하는 것은 비효율적이고 오류 발생 가능성이 높기 때문에, 자동화된 도구가 요구됨

· 이러한 필요에 따라 DBPTK가 개발됨. 이 도구는 오픈소스 소프트웨어로 누구나 자유롭게 사용할 수 있으며, 다양한 데이터베이스 포맷 간 변환을 자동으로 처리해 줌

· DBPTK는 모듈형 아키텍처를 채택하여, '입력 모듈'과 '출력 모듈'을 조합하여 데이터베이스 간 형식 변환이 가능함

  ◦ 입력 모듈(Import Module): 데이터베이스에서 메타데이터와 실제 데이터를 추출하는 기능을 담당함

  ◦ 출력 모듈(Export Module): 추출된 정보를 특정 포맷 또는 DBMS 형식으로 저장함

· 이처럼 DBPTK는 서로 다른 데이터베이스 시스템 간의 변환은 물론, 보존포맷에서 다시 DBMS로의 복원도 가능함

· 주요 지원 DBMS는 <표 8>과 같음

<표 8> DBPTK의 지원 DBMS

DBMS	입력 지원	출력 지원
Oracle	○	○
MySQL	○	○
Microsoft SQL Server	○	○
PostgreSQL	○	○
Microsoft Access	○	X

- Database Preservation Toolkit의 파일럿 테스트 사례

· DBPTK는 E-ARK 프로젝트의 일환으로 개발되었고, 다양한 유럽 국가 기록기관에서 실제 운영환경을 기반으로 한 파일럿 테스트를 통해 검증함

· DBPTK의 변환 품질을 검증하기 위해, 신뢰성과 정확성을 확인할 수 있는 전용 테스트 시스템이 개발되었고, 이 시스템은 '라운드트립 테스트(roundtrip testing)'라 불림

· 테스트 절차는 먼저, 원본 데이터베이스를 다른 포맷으로 변환함. 변환된 결과를 다시 원래 포맷으로 재변환하고, 변환 전과 후의 데이터를 비교하여 손실 여부나 변형이 있는지 확인함

· <표 9>는 DBPTK가 다양한 DBMS 환경 및 복잡한 데이터 구조에 대한 적응력을 검증받았고, 변환 정확성과 보존 가능성이 입증되었음을 보여줌. 특히 대용량 데이터세트, 비정규화 구조, 민감 정보 포함 등 다양한 특성을 갖는 데이터베이스에 대해서도 안정적인 성능을 보임

<표 9> 국가별 파일럿 테스트 사례

기관	데이터베이스 정보	DBMS	주요 내용
덴마크 국립기록원	덴마크 전염병 보고 시스템	MS SQL Server	90개 테이블, 50만 건
	민간기업 Kultunaut ApS.의 문화행사 정보	MySQL	5개 테이블, 3,300만 건
	과학 연구데이터 수신 시스템	MS SQL Server	289개 테이블, 130만 LOB
	고등교육부 상담 기록	MS SQL Server	180개 테이블, 10만 LOB
헝가리 국립기록원	헝가리 검찰청 사례 기록 등	Oracle	비정규화 DB 포함, 수십만 건
에스토니아 국립기록원	복잡한 구조의 DB	미상	20만 건
슬로베니아 국립기록원	MS Access DB	MS Access	SIARD2로 마이그레이션, 도구 개선 피드백 제공

- 평가 및 분석

· DBPTK는 Oracle, MySQL, PostgreSQL, MSSQL 등 주요 DBMS와의 데이터 입출력을 지원하며, 모듈형 구조로 설계되어 새로운 DBMS도 손쉽게 확장 가능함. MS Access는 입력만 지원함. 이와 같은 모듈 기반 아키텍처는 다양한 환경에서의 상호운용성을 강화하는 핵심 요소임

· 텍스트 형태의 대용량 데이터(LOB)는 XML 내에 직접 기록할 수 있으며, 바이너리 대형 객체(BLOB)는 SIARD에서 직접 저장하지 않음. 대신 보존포맷 설계 시 URI를 활용해 외부 저장소의 대용량 파일을 연결하는 방식을 적용할 수 있어 저장 효율성과 시스템 간 확장성 확보에 유리함

· DBPTK는 전체 데이터베이스를 변환하지 않고, 특정 테이블·열·행을 선별하여 보존포맷으로 추출 및 변환하는 기능을 지원함. 이는 SIARD 2.0의 테이블 단위 보존 기능을 구현한 것으로, 보존 범위를 유연하게 조정할 수 있어 실무 아카이빙 환경에서의 활용도가 높음

· SIARD 2.0는 스위스 연방 기록원과 E-ARK 프로젝트가 개발한 국제표준 포맷으로, XML과 SQL 표준을 기반으로 함. 이로써 다양한 시스템과의 높은 상호운용성을 보장하며, 메타데이터 안에 URI를 기록하여 외부 저장소의 파일을 참조하는 오픈 포맷·국제표준 기반 설계를 가능하게 함

나. 기록분야 이외 데이터세트 보존포맷 및 장기보존패키지의 현황 조사 및 분석

○ **(기업/산업)** SAP의 SAP ILM

- 개요

· SAP의 ILM(Information Lifecycle Management)은 IT 인프라를 효율적으로 정비할 수 있도록 지원하고, 기존 시스템의 폐기 절차를 간소화하며 데이터 보존 관리를 자동화하는 기능을 제공함

· 사용자는 데이터를 검색할 수 있는 형식으로 아카이빙할 수 있고, 아카이빙된 데이터에 대한 감사 및 보고 기능과 규칙 기반 보존 정책의 중앙 집중식 관리 및 운영을 통해 비즈니스 리

스크를 최소화하고 비용을 절감하며 통제력을 강화할 수 있음

· 데이터 아카이빙, 보존 관리, 시스템 및 데이터 폐기 기능을 종합적으로 제공함으로써, 기업들은 SAP ILM을 통해 효율적이고 안전한 데이터 관리를 실현하고, 지속 가능한 데이터 거버넌스 체계를 구축할 수 있음

- 관리 기능(<그림 12> 참조)

· Audit Areas: 어떤 데이터가 어떤 규칙에 따라 보존되고 삭제되는지 구분함

· ILM Rules: 데이터의 보존기간, 삭제 시점 등을 정의한다. 법적·업무적 요구사항을 반영함

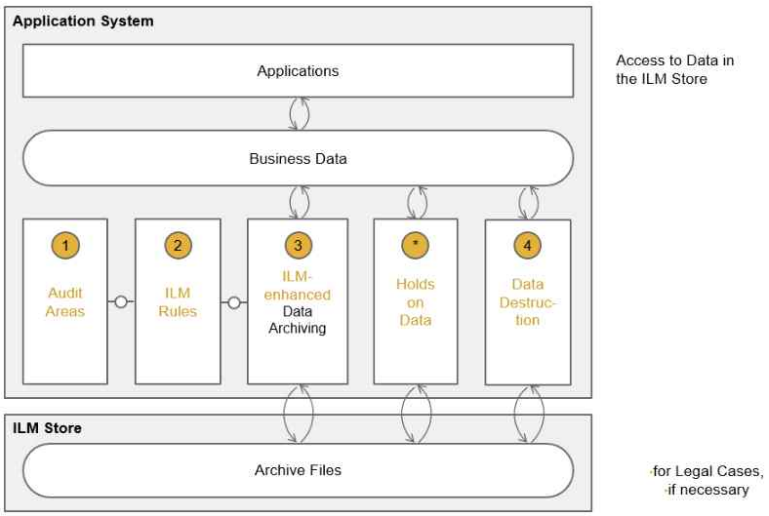
· ILM-enhanced Data Archiving: 정의된 규칙에 맞춰 데이터를 압축·저장하고 필요시 검색 가능하도록 보존함

· Holds on Data: 법적 분쟁이나 감사 등 특별한 이유로 데이터를 삭제하지 않고 '보류' 상태로 유지하는 기능임

· Data Destruction: 보존기간이 끝나면 규칙에 따라 데이터를 안전하게 삭제함

· Archive Files: 아카이빙된 데이터 파일이 저장되는 공간임

· 시스템 폐기: SAP ECC 등, 더 이상 사용하지 않는 시스템을 폐기하면서도 해당 시스템에 저장된 데이터를 ILM 보존 웨어하우스 시스템(RW 시스템)으로 이관하여 접근성을 유지함



<그림 12> 운영 시스템 내 보존 관리 기능

- 평가 및 분석

· 스캔 문서나 PDF 파일은 SAP ArchiveLink 기능을 통해 함께 보관할 수 있음. 하지만 데이터베이스 안에 있는 이미지 및 영상과 같은 외부 파일을 직접 아카이빙하는 방식은 정확히 서술되어 있지 않음

· 테이블에서 필요한 일부만 빼내서 저장하는 것은 어려움

· SAP만의 자체 저장 형식을 사용하며, PDF, CSV 같은 오픈 포맷 지원은 부족함

· SAP는 법적 규제 준수, 감사 추적, 보존 및 폐기 정책 이행과 같은 컴플라이언스 중심 기능 제공에 강점을 가짐

○ **(기업/산업)** Exsoft(엑스소프트)의 eXrep/SAP BizConnector

- 개요

· eXrep/SAP BizConnector는 순수 국내 기술로 개발된 최초의 SAP 아카이빙 솔루션으로, SAP 시스템에서 생성·등록되는 다양한 문서를 ECM(통합 콘텐츠 관리 저장소)에 아카이빙함으로써 SAP 성능 향상과 업무 효율성 증대를 지원함

- 실행 구조(<그림 13> 참조)

· SAP Native UI: 사용자가 SAP 애플리케이션을 사용하는 화면임

· SAP Applications & Business Object: SAP 내에서 생성되는 다양한 업무 문서(비즈니스 오브젝트, 출력물, ADK 파일 등)를 처리함

· ArchiveLink: SAP의 표준 아카이빙 인터페이스로, 문서와 데이터가 외부 저장소(eXrep ECM)와 연결되는 경로임

· SAP Database: 운영 데이터가 저장되는 DB임

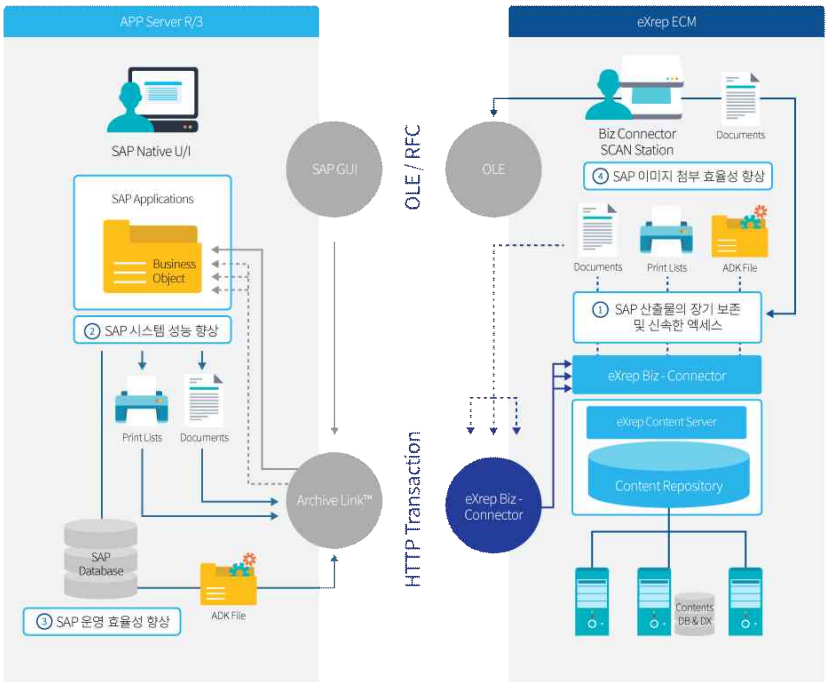
· ADK File: SAP 아카이빙 데이터 파일임

· BizConnector SCAN Station: 종이문서나 이미지 자료를 스캔하여 디지털 파일로 변환하고, SAP 문서와 연계 저장함

· Documents, Print List, ADK File: SAP에서 넘어온 문서·리포트·아카이빙 파일을 저장함

· eXrep BizConnector: SAP와 eXrep ECM 간 데이터 전송을 담당하는 핵심 모듈임

· Content Repository: ECM의 핵심 저장소로, 아카이빙된 모든 콘텐츠(문서·리포트·이미지 등)를 보관함



<그림 13> SAP 문서관리시스템 실행 구조

- 특징 및 장점

· 통합 ECM 기반 SAP 아카이빙 솔루션으로 SAP에서 발생하는 다양한 문서(비즈니스 문서,

ADK 파일, 프린트 리스트 등)를 통합 콘텐츠 관리 저장소에 아카이빙하여 SAP 시스템 성능과 업무 효율성을 향상시킴

· 형식이 서로 다른 모든 자료를 ECM 저장소에서 같은 방식으로 관리할 수 있음. 대용량 파일이나 다수의 사용자가 있어도 안정적으로 운영할 수 있고, 이후에 데이터 증가 시에도 시스템 확장이 용이함

· SAP 성능 및 운영 효율성을 향상시킴. SAP 아카이빙 표준을 활용해 산출물을 ECM에 별도로 보관 및 관리함으로써 통합 관리와 즉각적인 조회 서비스를 제공함

· 고객 요구에 따라 국내 R&D 기술지원 인력을 통한 엔진 레벨의 커스터마이징을 지원함. 또한 Framework 기반의 개발 환경을 통해 요구사항을 신속하게 반영할 수 있으며, 기존 SAP 시스템에서 관리가 어려웠던 종이문서에 대해서도 스캔 및 바코드 인식 등 종합적인 이미지 처리 모듈을 제공함

- 주요 기능: Early Archiving(<그림 14> 참조)

· SAP에서 거래나 업무 기록이 만들어지기 전에 문서를 먼저 저장하는 방식임

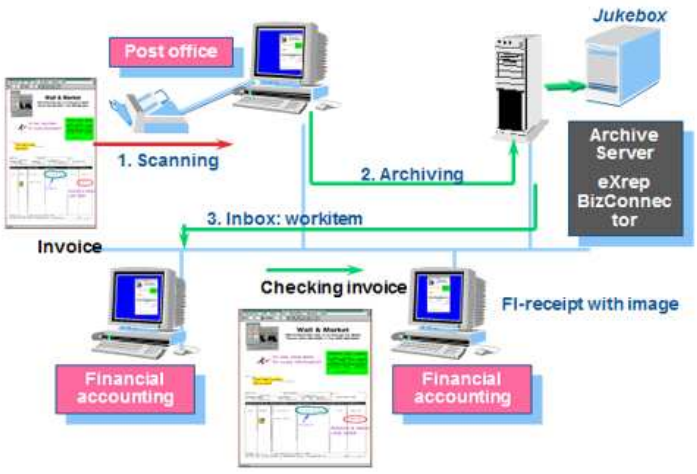
· 1단계 스캔(Scanning): 우편이나 다른 경로로 송장이 도착하면, 먼저 스캐너로 이미지를 만듦. 스캔 시 바코드나 문서 번호 같은 정보가 함께 인식됨

· 2단계 아카이빙(Archiving): 스캔된 이미지 파일이 eXrep BizConnector를 통해 아카이브 서버에 저장됨. 이렇게 하면 원본 문서는 안전하게 보관되고, 필요할 때 검색 가능함

· 3단계 워크아이템 전달(Inbox: workitem): 아카이빙된 문서 정보가 SAP 워크플로우를 통해 재무회계 담당자의 업무함(Inbox)으로 전달됨

· 4단계 송장 확인(Checking invoice): 회계 담당자가 SAP에서 송장을 확인하고 처리함. 이때 시스템은 이미 저장된 스캔본과 거래 정보를 자동으로 연결함

· 5단계 이미지가 포함된 회계 전표(FI-receipt with image): 최종적으로 회계 전표 화면에서 해당 송장의 이미지가 함께 조회됨



<그림 14> Early Archiving

- 주요 기능: Late Archiving(<그림 15> 참조)

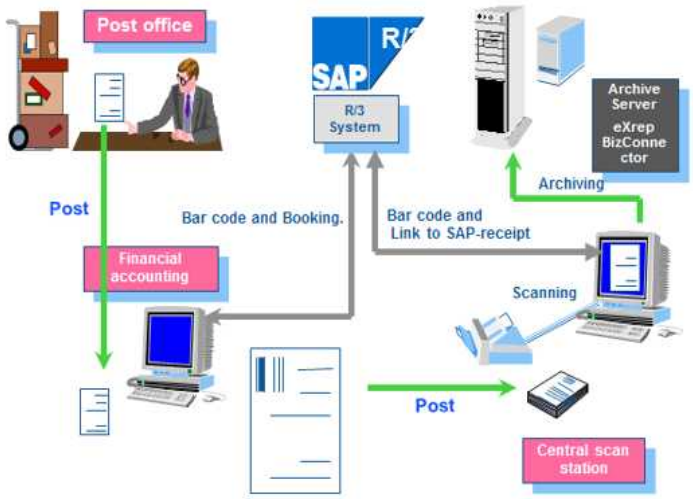
· 1단계 문서 접수 및 회계 처리: 우편 등으로 송장이 도착하면, 재무회계 담당자가 SAP 시스템에 내용을 먼저 입력(Booking)하고 바코드를 부여함. 이 바코드에는 해당 거래의 고유 번호가 담겨 있음

· 2단계 문서 스캔: 문서는 중앙 스캔 스테이션으로 보내져서 스캔됨. 스캔 시 문서에 붙은 바

코드가 함께 인식되어, 어떤 거래와 연결될지 자동으로 식별됨

· 3단계 아카이빙: 스캔본은 eXrep BizConnector를 통해 아카이브 서버에 저장됨. 바코드 정보 덕분에, 저장 시 해당 스캔본이 SAP에 이미 입력된 거래 기록과 자동 연결됨

· 4단계 조회 가능: 이후 SAP에서 해당 거래를 조회하면, 입력된 데이터와 함께 스캔된 원본 이미지도 바로 확인할 수 있음



<그림 15> Late Archiving

- 주요 기능: Outgoing Document Archiving(<그림 16> 참조)

· 외부로 전달되는 문서의 경우 SAP Script의 형태로 만들어진 폼(form)에 의해 출력되며, Output Determination에 의해 PDF 형식의 Archiving 파일로 생성함

· 기존에는 수동으로 처리하던 문서를 자동으로 아카이빙하여 저장 및 관리함

· 종이문서를 별도로 저장할 필요가 없으며, 언제 어디에서나 조회 가능함

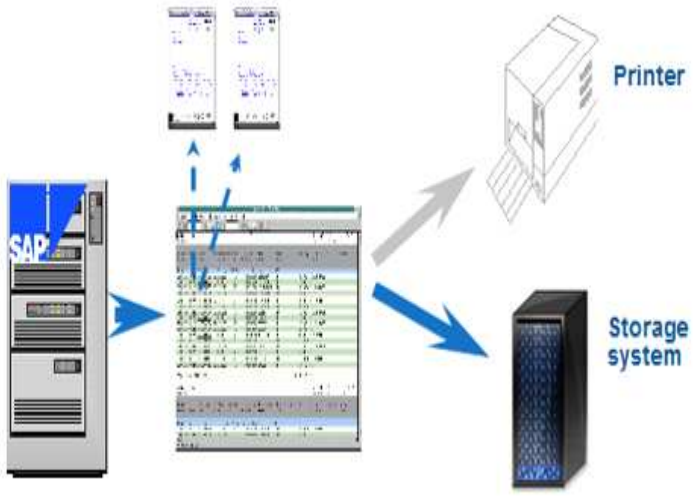
<그림 16> Outgoing Document Archiving

- 주요 기능: Print List Archiving(<그림 17> 참조)

· SAP에서 발생한 Report를 아카이빙하여, 어디에서든 쉽게 조회 가능함

· Print List Archiving할 때의 출력물이 그대로 보존됨

· Print List Index를 이용하여 효과적이고 신속한 검색이 가능함



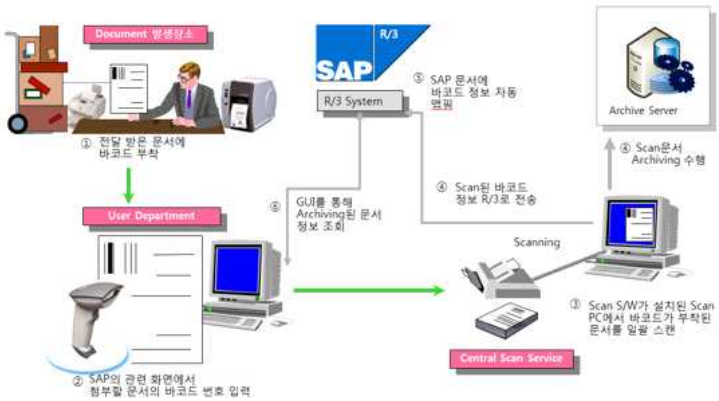
<그림 17> Print List Archiving

- 주요 기능: 종이문서 바코드 스캔 기능(<그림 18> 참조)

· 바코드 인식을 통해 대량의 종이문서를 빠르고 편리하게 등록할 수 있음

· 종이문서에 바코드를 부착하여 스캔한 이미지를 BizConnector를 통해 통합 저장소에 저장하고 관리할 수 있음

· 사용자는 SAP에서 바코드 번호만을 입력함으로써, 편리하고 손쉽게 저장된 이미지를 첨부할 수 있음



<그림 18> 종이문서 바코드 스캔 기능

- 평가 및 분석

· SAP ArchiveLink 표준을 기반으로 하여 SAP에서 생성된 트랜잭션 중심 문서만 아카이빙 가능함. SAP 전용 환경에 최적화되어 있어 타 DBMS 기반의 관계형 데이터는 직접적으로 수

용하지 못함

· 스캔 이미지, ADK 파일, PDF 등 비정형 콘텐츠 저장 기능을 제공하며, 바코드 기반 이미지 연결 기능을 통해 문서와 거래 데이터를 자동 매칭할 수 있음

· 문서 단위 보존에 특화되어 있어 트랜잭션 문서나 출력 리스트 수준의 아카이빙은 가능하지만, 데이터베이스 테이블 단위의 구조적 보존은 지원하지 않음

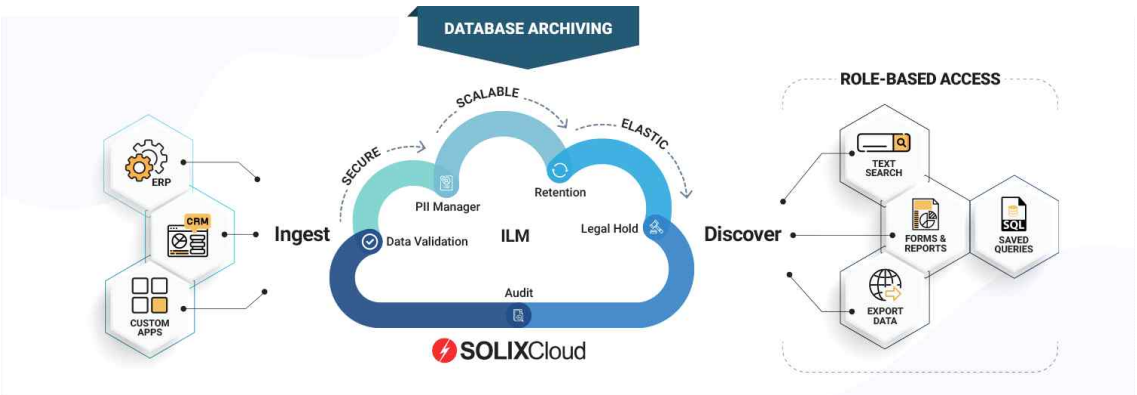
· 자체 저장 포맷을 사용하며, PDF·CSV 등 오픈 포맷 지원은 제한적임

○ **(기업/산업)** Solix Tech.의 SOLIXCloud

- 개요

· SOLIXCloud의 데이터베이스 아카이빙은 기업의 시스템에서 증가하는 데이터를 효율적으로 관리하고, 멀티 클라우드 기반의 아카이빙 솔루션을 통해 데이터 유지·관리 비용을 절감할 수 있도록 지원함

· 정보 수명 주기 관리(ILM) 정책과 규칙을 적용하여 데이터 이관·검증·폐기를 자동화하고, 관계형 데이터베이스와 기업의 소프트웨어를 폭넓게 지원함



<그림 19> 데이터베이스 아카이빙 개요

- 주요 특징

· Oracle, Mainframes, SQL Server, DB2, Sybase, Teradata, MySQL 등의 관계형 데이터베이스를 지원함

· 저렴한 비용의 클라우드 및 온프레미스 스토리지를 지원함

· 고객, 제품, 부서 같은 기본 정보를 기준으로, 서로 다른 테이블에 있는 관련 데이터를 맞춰서 함께 보관할 수 있음

· SAP, 오라클 E-비즈니스, 피플 소프트, 시벨, JD 에드워즈 등과 같은 유명 기업용 소프트웨어에 맞춘 사전 구축 지식을 기반으로 함

· 사용자가 직접 필요한 데이터를 검색하고 조회할 수 있음

- 주요 기능

· 모든 형태의 데이터 지원: SOLIXCloud Enterprise Archiving은 구조화(표로 구성된 파일), 반구조화(이메일 및 로그 파일 등), 비구조화(이미지와 영상 등) 파일과 기존에 사용하였으나 현재는 사용하지 않는 소프트웨어를 안전하게 폐기할 수 있도록, 클라우드 환경에서 관리되는 대규모 아카이빙 저장소를 제공함

· 법·규제 준수: 공정노동기준법, GDPR, HIPAA 등 다양한 국내외 법규에 맞춰 데이터 보존 기간을 관리하고, 기간이 지나면 자동 또는 수동으로 삭제할 수 있음. 기업마다 다른 보존 규칙을 직접 설정할 수도 있음

· 삭제 방지: 데이터는 법적으로 정해진 기간 동안 안전하게 보관되고, 삭제 전에 알림이 울림

· 법적 요청 대응(eDiscovery, Legal Hold): 소송이나 조사 시, 필요한 데이터를 빠르게 찾아내고, 보존기간이 끝나도 삭제되지 않도록 잠금(Legal Hold) 기능을 걸 수 있음

· 가상 프린터 기능(Solix Virtual Printer): 오래된 시스템에서 보고서를 인쇄하는 대신 바로 아카이빙 저장소로 보낼 수 있어, 나중에 검색과 활용이 쉬움

· 강력한 검색(Solix Search): 문서 안의 글자까지 검색할 수 있고, 보고서나 분석 쿼리도 즉석에서 만들 수 있음. 예를 들어, 송장이나 거래 내역을 바로 찾아볼 수 있음

· 시스템 성능·비용 최적화: 자주 쓰지 않는 데이터를 따로 보관해 운영 시스템 속도를 높이고 저장 비용을 줄임

· 메타데이터 관리: 기업 전반의 데이터 흐름과 관계를 한눈에 파악하고, 모든 데이터에 일관된 설명(메타데이터)을 붙여 체계적으로 관리할 수 있음

· 다양한 연동·보고 지원: ODBC, JDBC, SQL, REST API로 다른 시스템과 연결 가능하며, 기존 보고서나 즉석 보고서도 생성 가능함

- 평가 및 분석

· Oracle, SQL Server, MySQL, DB2, Teradata 등 여러 DBMS를 사용할 수 있고, SAP·Oracle EBS·PeopleSoft·JD Edwards 같은 주요 ERP 시스템과도 잘 연동됨

· 텍스트 보고서, 반구조화 문서, PDF, ERP 출력 보고서 같은 형식도 Virtual Printer 기능을 이용해 쉽게 수집하고 저장할 수 있음

· 이미지, 영상, 음성 파일처럼 비정형 데이터도 저장할 수 있음. 저장 시 파일 자체를 원본 그대로 보관할 수 있고, 파일 내용에 대한 메타데이터(파일명, 날짜, 설명, 관련 거래 번호 등)를 함께 기록해 검색과 관리가 가능함

○ **(기업/산업)** CSP의 CHRONOS

- 개요

· CHRONOS는 산업 현장에서 사용되는 소프트웨어의 데이터베이스 보존 솔루션으로, 사용이 중단된 소프트웨어와 데이터베이스를 장기간 안전하게 보관하도록 지원함

· 일부 데이터를 계속 운영하면서, 선택한 데이터만 장기보존할 수 있음

- 주요 특징

· CHRONOS에 보존된 데이터는 일반 데이터베이스처럼 SQL을 써서 검색·조회할 수 있고, 그 결과로 보고서를 만들 수 있음

· CHRONOS가 데이터를 텍스트 파일 형태로 저장하지만, 사용자가 조회할 때는 일반 데이터베이스처럼 빠르고 편하게 SQL을 써서 검색할 수 있음

· CHRONOS는 1차 데이터(primary data)뿐 아니라 테이블, 뷰, 인덱스, 패키지, 프로시저, 함수, 트리거, 시퀀스, 구체화된 뷰(Materialized View), 스케줄러, 검사 제약조건(Check Constraints)도 함께 보존함

· CHRONOS에서 데이터를 옮길 때 단순히 데이터를 주고받는 기능인 큐(Queues), 다른 DB와 연결하는 기능인 데이터베이스 링크(Database Links), DB 밖에서 관리되는 사용자·권한 정보는 보존 대상에서 제외됨

· 데이터베이스 구조 변경(예: 새로운 컬럼 추가)을 자동으로 감지하여 새 버전으로 저장함

· 값의 의미 변경(예: 코드 체계 변경)은 자동 감지가 불가능하므로 사용자가 변환 규칙을 정하면 전체 데이터에 적용됨

· 실제 데이터는 그대로 두고, 변환은 CHRONOS 안에서만 반영함

· JDBC, Java RMI, 웹서비스 등을 통해 개발자가 직접 접근 가능함

· 데이터 내용과 구조를 분리하여, 구조는 XML·XSD 파일로 저장해 어떤 시스템에서도 복구할 수 있게 함

· 테이블 간 연결 관계를 자동으로 찾아주고, 순환 참조나 외부 시스템 연결도 관리할 수 있음

- 주요 기능

· CHRONOS는 내보낸 데이터에 법적 보존 정책을 적용할 수 있는 모듈을 제공하며, 다양한 저장 매체와 분산 아카이브를 중앙에서 관리함. EMC 등 전문 아카이브 시스템과 연동이 가능하고, 일반 파일 아카이브에도 수동으로 보존 전략과 만료일을 설정할 수 있음. 데이터 삭제는 아카이브 패키지 단위로 2단계 승인 절차를 거쳐 안전하게 수행됨

· CHRONOS는 Oracle의 사용자 정의 데이터 타입(UDT)을 아카이빙할 수 있으며, 데이터 추출 시 DBMS별 고유 SQL을 사용하고, 조회 시에는 SQL92 표준 인터페이스를 제공함. 또한 Oracle Forms 기반 애플리케이션과는 자체 API를 통해 연동이 가능함

· 모든 구성요소에 통합된 접근 제어·사용자 관리 기능을 제공하며, LDAP 연동이 가능하고, 컬럼 단위까지 세밀한 권한 설정을 지원함

· CHRONOS는 파일 시스템에 있는 아카이브 데이터에 직접 SQL을 실행할 수 있으며, SQL92 인터프리터·전역 검색 인덱스·컬럼 단위 BTree 인덱스·인메모리 DB 등을 결합한 하이브리드 방식으로 표준 DB 수준의 성능을 제공함

- 평가 및 분석

· XML 기반 구조와 SQL 질의가 가능한 텍스트 파일을 저장하여 특정 데이터베이스 제품에 묶이지 않음

· 비구조화 파일(예. 이미지, PDF, ERP 보고서 출력물 등)은 주로 Virtual Printer 기능이나 파일 직접 수집 방식을 통해 저장됨. 따라서 원본 그대로 파일 시스템에 저장되고, 검색과 관리를 할 수 있도록 메타데이터가 함께 기록됨. 이 메타데이터는 어떤 데이터세트나 거래와 연결되는지 참조 정보를 지원함

· 데이터를 옮길 때 필요한 테이블, 뷰, 인덱스 등 원하는 것만 골라서 내보낼 수 있음

○ **(기업/산업)** OpenText(오픈 텍스트)의 InfoArchive

- 개요

· InfoArchive는 데이터 아카이빙 솔루션으로, 기업이 보유한 방대한 데이터를 안전하고 효율적으로 보존하고 활용할 수 있도록 지원함

· 관계형 데이터베이스 같은 정형 데이터와 문서, 이미지, 동영상 등 비정형 데이터를 하나의 통합 환경에서 장기 보존할 수 있는 것이 특징임

· 사용하지 않는 시스템 내에 남아 있는 데이터를 사용자가 설정한 법적 기준과 업무 규정에 맞춰 안전하게 이관할 수 있음

· 사용자별 접근 권한 관리, 데이터 암호화 및 마스킹, 접근 기록, 개인정보 보호 기능을 제공함

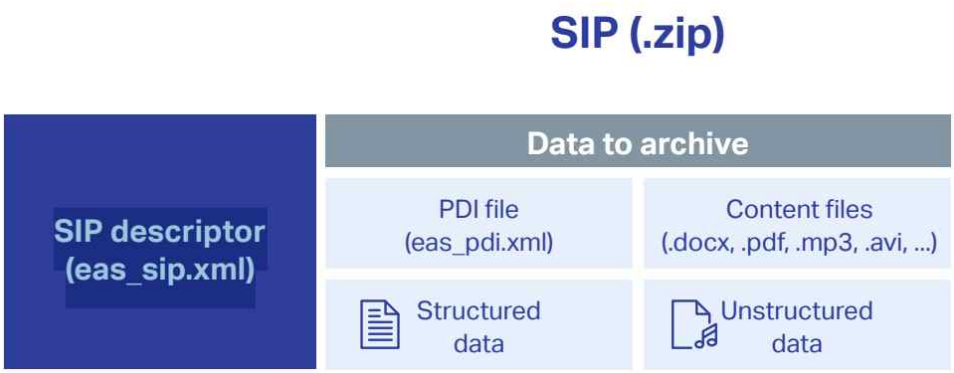
· 하나의 관리 화면에서 모든 데이터를 통합 관리할 수 있고, 직관적인 UI를 통해 효율적인 운영이 가능함

· 다른 분석 프로그램이나 외부 서비스와 연결할 수 있어, 보관된 데이터를 분석하고 활용할 수 있음

- 시스템 구조

· InfoArchive의 시스템은 OAIS(Open Archival Information System) 참조모델을 기반으로 설계되어, 데이터의 제출(SIP), 보존(AIP), 제공(DIP) 단계에 따라 관리 및 운영되도록 구성되어 있음

· SIP(Submission Information Package)의 구조는 다음과 같음(<그림 20> 참조)



<그림 20> SIP의 구조

· SIP 기술서(Descriptor): SIP 자체를 설명하는 메타데이터 파일임. 이 패키지 안에 무엇이 들어있는지, 어떤 데이터인지 기본 정보를 기록함

· 보존 데이터(Data to archive): 정형 데이터와 비정형 데이터를 논리적 관리 단위로 결합하여 보존함. 이때 크게 두 가지 요소로 구성됨. PDI 파일은 정형 데이터를 기술하는 XML로, 데이터베이스에서 추출된 표 구조를 설명함. 콘텐츠 파일은 비정형 데이터로, 문서·이미지·음성·영상 등의 파일이 별도로 보존되며, PDI와 URI 연결을 통해 논리적으로 연계됨

· AIP(Archival Information Package): SIP가 보존을 위해 변환된 형태로, 데이터 무결성과 장기보존을 보장함

· DIP(Dissemination Information Package): 사용자가 데이터를 요청할 때 AIP에서 필요한 데이터를 추출하여 제공하는 패키지임

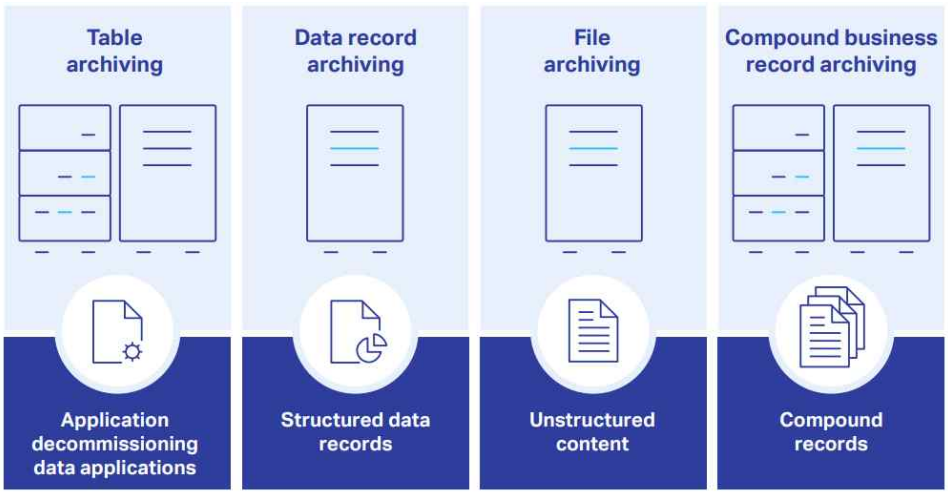
- 아카이빙 방식(<그림 21>)

· 테이블 아카이빙(Table archiving): 정형 데이터를 저장하는 방식으로, 데이터베이스를 XML 형태로 저장하고, 테이블·열·행 단위로 재현할 수 있음

· 데이터 레코드 아카이빙(Data record archiving): 정형 데이터를 특정 단위로 쪼개 저장하고, 연관된 첨부파일(PDF, 이미지 등)은 URI를 통해 연계함

· 파일 아카이빙(File archiving): 문서, 이미지, 동영상과 같은 비정형 데이터를 파일 단위로 보존하는 방식임. 파일은 원본 그대로 또는 장기보존 포맷(PDF/A 등)으로 변환하여 저장하며, 메타데이터와 함께 논리적으로 관리됨

· 복합 레코드 아카이빙(Compound business record archiving): 정형 데이터와 비정형 데이터를 하나의 논리적 레코드 단위로 결합해 관리하는 방식임. 메타데이터와 URI 참조를 통해 연결하여 특정 이벤트와 관련된 모든 데이터를 하나의 비즈니스 문맥에서 통합적으로 이해할 수 있도록 함



<그림 21> InfoArchive의 운영 옵션

- 보안 및 거버넌스
  · 접근 권한 설정: 사용자의 역할에 따라 볼 수 있는 데이터를 제한할 수 있어, 허가받지 않은 사람이 중요한 정보에 접근하지 못하도록 할 수 있음
  · 데이터 보호: 데이터를 암호화하거나 일부 정보를 가려서 보여줄 수 있어 개인정보 유출을 막을 수 있음
  · 기록 추적: 누가 언제 어떤 데이터에 접근했는지 기록을 남겨 관리할 수 있음
  · 데이터 삭제 방지: 법적 문제나 규정상 보관이 필요한 데이터는 실수나 의도적으로 삭제하지 못하도록 할 수 있음
  · 확장성: 페타바이트(Petabyte) 규모의 대규모 데이터 보관이 가능함
- 구축 및 서비스 형태
  · 온프레미스(On-Premises): 회사나 기관에서 보유한 자체 서버에 설치하여 운영하는 방식임
  · 퍼블릭 클라우드(Public Cloud): 온라인 기반의 공용 클라우드 서비스를 활용하여 운영하는 방식임
  · 프라이빗 클라우드(Private Cloud): 설치와 유지보수를 OpenText 또는 인증된 기업이 대신 관리해주는 서비스 형태임
- 평가 및 분석
  · 데이터베이스 기반의 정형 데이터뿐 아니라 문서, 이미지, 음성, 동영상 등 비정형 데이터까지 하나의 플랫폼에서 장기보존 가능함
  · Oracle, SQL Server, DB2 등 주요 데이터베이스와 연동이 가능하며, 특정 테이블·행·열 등 필요한 데이터만 선택적으로 보관할 수 있음
  · 복합 레코드 아카이빙 기능을 통해 사건·업무 단위로 관련 데이터를 메타데이터와 URI로 논리적으로 연결함으로써 기록의 맥락을 유지한 채 관리할 수 있음. 이는 DB와 파일을 패키징하여 보존하는 구조 설계와 유사한 방식으로, 복합적 기록관리에 적합함
  · 비정형 데이터는 원본 그대로 또는 PDF/A, TIFF 등 장기보존 표준 포맷으로 변환하여 저장할 수 있어 향후 접근성 및 상호운용성을 높일 수 있음

○ **(과학기술)** Fermilab(페르미 국립 가속기 연구소)의 R2DP 프로젝트

- Tevatron Run II에서 수집된 과학 데이터를 지속해서 활용할 수 있도록 데이터, 소프트웨어, 환경, 문서까지 종합적으로 보존하는 프로젝트임

- 데이터세트별 보존 방법

· 충돌 데이터(Collision Data): LTO4 테이프(용량 800GB)로 저장하였으나 LTO4 기술이 점차 산업계에서 사라지고 있어 향후 장비 교체 및 데이터 접근이 어려워질 위험이 있음. 따라서 T10K 테이프(최대 5TB)로 이관함

· 비통계 데이터(Non-statistical Data): Oracle 데이터베이스를 사용하여 관리함. 하지만 Oracle은 비용이 높은 상용 소프트웨어로 대부분의 최신 실험에서는 사용하지 않음. 하지만 PostgreSQL 같은 오픈소스 데이터베이스로 이관할 경우 막대한 인력과 비용이 소요됨. 이에 Oracle을 최신 버전으로 업그레이드한 뒤, 이후 버전 업그레이드로 인한 기존 소프트웨어 호환성 문제를 방지하기 위해 현재 버전과 스키마를 '동결'하였으며, 필요시 네트워크 격리 상태에서도 운영할 수 있도록 비상 계획을 수립함

- 소프트웨어 보존 전략

· 테바트론(Tevatron) 입자 가속기 실험은 CDF와 D0로 구분됨. 이 실험은 시뮬레이션, 재구성, 보정, 분석을 수행하는 소프트웨어 프레임워크를 보유하고 있고, 주로 32비트 Scientific Linux 환경에서 개발됨

· 종료 당시 운영 중이던 소프트웨어는 Scientific Linux 5(SL5) 기반이었지만, 오래된 라이브러리에 의존하고 있었기 때문에, CDF와 D0는 소프트웨어 기능 유지를 위해 각기 다른 전략을 채택해 대응함

<표 10> CDF와 D0의 소프트웨어 장기보존 전략

구분	CDF	D0
보존 전략	새로운 '레거시(legacy)' 릴리스 제작 (불필요한 패키지 및 구버전 라이브러리 제거)	기존 소프트웨어 릴리스를 유지, 공통 툴만 업데이트
운영체제 대응	Scientific Linux 5(SL5) 기준으로 정리 후, SL6에 쉽게 이식 가능하도록 준비	SL5 기반 유지, 필요 시 SL6 호환성 라이브러리 추가 설치 예정
주요 조치	레거시 릴리스 제작 및 검증 후, CVMFS에 배포	작업 노드에 32비트 호환 라이브러리 설치, 필요 시 CVMFS에 추가
소프트웨어 저장소 크기	약 326 GB	약 227 GB
특징	코드 간소화 및 향후 이식 용이성 확보	기존 시스템 유지하며 최소한의 수정으로 장기 지원

- 평가 및 분석

· 통계 수치 데이터뿐 아니라 로그 파일, 설정 파일 등 다양한 형식의 자료가 포함됨. CVMFS와 dCache와 같은 저장·전송 구조를 활용해 대용량 실험 데이터뿐 아니라 분석용 소프트웨어와 스크립트 파일에도 접근할 수 있음. 이는 단순 데이터 보관을 넘어 재현과 검증이 가능한 연구데이터 관리 구조로 설계된 특징임

· 데이터베이스 스키마는 동결되어 변경되지 않으며, PostgreSQL 전환 후에는 SQL 쿼리를 통해 개별 테이블 단위 데이터 조회·추출이 가능함. 오픈소스 DBMS(PostgreSQL)를 사용해 접근성과 장기적 활용성을 보장함

· DBMS로 사용하는 PostgreSQL은 오픈소스이며, Oracle은 상용이지만 스키마 동결로 접근성을 유지하고 있음. 데이터 저장 매체인 T10K 테이프는 물리적 포맷 자체보다 접근·읽기 방식이 더 중요한 요소임

· 로그 파일, 분석 스크립트, 실행 환경 설정 파일, 문서(PDF 등) 등 비구조화 데이터는 파일시스템에 저장하고, 데이터베이스에는 경로·버전·해시값 등을 기록해 구조화 데이터와 연결함으로써 실험 데이터, 코드, 환경을 통합적으로 관리할 수 있음

○ **(과학기술)** ETH Zurich(취리히 연방 공과 대학교)의 ETH Data Archive

- 개요

· ETH Data Archive는 ETH Zurich의 디지털 데이터를 장기적으로 보존하기 위한 시스템임

· ETH Zurich 소속 구성원의 연구데이터는 ETH Data Archive에 자동으로 이관되어 장기보존됨

· ETH Data Archive는 OAIS 표준을 기반으로 장기보존을 위한 절차를 정의하고 있으며, PREMIS를 기반으로 장기보존에 필요한 메타데이터를 구현하고 있음

<표 11> ETH Data Archive의 보존 포맷

파일 포맷	권장 포맷	제한된 범위에서만 허용	적합하지 않은 포맷
서식이 있는 텍스트	· PDF/A · XML	· 폰트가 포함된 PDF(*.pdf) · PDF/A-3(*.pdf) · RTF(*.rtf) · HTML 및 XML(ASCII 기반의 텍스트 형식으로 장기적으로 접근이 가능함. 단, 외부 링크는 가급적 피할 것) · 첨부파일의 경우 허용되는 형식 · Word(*.docx) · PowerPoint(*.pptx) · LaTeX, TeX(ASCII 텍스트 형식으로 장기 보존이 가능하지만, 오픈소스 소프트웨어가 필요하며 최종 PDF 결과물도 함께 포함해야 함) · 오픈 포맷 (*.odm, *.odt, *.odg, *.odc, *.odf) · 마크다운(*.md)	· Word(*.doc) · PowerPoint(*.ppt)
일반 텍스트	· 일반 텍스트(*.txt, *.asc, *.c, *.h, *.cpp, *.m, *.py, *.r etc.) · 마크다운(*.md)	·	
스프레드시트 또는 표	· 쉼표(,) 또는 탭(tab)으로 구분된 텍스트 파일(*.csv)	· Excel(*.xlsx) · 오픈 포맷(*.ods)	· Excel(*.xls, *.xlsb(바이너리 형식))

원시 데이터 및 워크스페이스		· ASCII 텍스트 형식이 장기 보존에 적합하지만, 데이터를 가져오는 시간이 오래 걸릴 수 있음 · S-Plus 파일 (*.sdd) · Matlab(*.mat)파일은 HDF 형식으로 저장 가능 (단, ASCII 형식으로 저장된 복잡한 *.mat 파일은 Matlab의 Load 명령어로 읽을 수 없기 때문에 피하는 것을 권장) · NetCDF(*.nc, *.cdf) · HDF5 (*.h5, *.hdf5, *.he5)	· *.mat 형식의 구버전 Matlab 파일이나 .RData 형식의 R 파일과 같은 바이너리 파일
래스터 이미지	· TIFF(*.tif) · PNG(*.png) · JPEG2000(*.jp2) · 디지털 네거티브 포맷 (*.dng, 디지털 사진의 원본 데이터를 보존하기 위해 사용되며, TIFF 형식의 사본과 함께 저장하는 것을 권장함)		· Photoshop(*.psd)
벡터 그래픽	· JavaScript와 연결 없이 순수 그래픽만 포함된 SVG 파일(*svg)		· InDesign (*.indd) · Illustrator (*.ait) · Encapsulated Postscript (*.eps)
CAD	· AutoCAD 도면파일(*.dwg) · AutoCAD용 도면 변환 파일 형식(*.dxf) · 3D, X3D 확장 파일 형식 (*.x3d, *.x3dv, *.x3db)		
오디오	· WAV(*.wav, 무압축·PCM 형식)	· AAC 오디오를 포함한 MPEG-4 형식(*.mpg4) · MP3(*.mp3)	
비디오	· FFV1 codec (Matroska 컨테이너(*.mkv) 안에 저장된 버전 3이상)	· MPEG-2(*.mpg,*.mpeg) · MP4(*.mp4) · QuickTime Movie(*.mov) · AVI (*.avi)	· Windows Media Video(*.wmv)

- 평가 및 분석

· CSV, JSON, XML, NetCDF, HDF5 등 범용 오픈 포맷을 지원해 다양한 시스템에서 데이터를 수용하고 장기보존 가능함

· 텍스트 기반의 대형 데이터는 XML, TXT 등 표준 텍스트 포맷으로 저장할 수 있으며, 바이너리 객체의 경우 TIFF, WAV, MKV 등 국제적으로 통용되는 표준 포맷을 권장 포맷으로 지정해 장기보존 가능함

· 보존 정책에서 오픈 포맷 사용을 원칙으로 하며, 모든 권장 포맷은 공개 표준 기반임. 독점 포맷은 사용을 제한하거나 장기보존 포맷으로는 권장하지 않음

○ **(과학기술)** Digital Science의 Figshare

- 개요

· Figshare는 연구자들이 데이터, 소프트웨어, 포스터, 논문 등 다양한 연구 산출물을 저장하고 공개할 수 있도록 지원하는 연구데이터 저장 및 공유 플랫폼임. 사용자는 데이터를 안전하게 보관하고, FAIR(Findable, Accessible, Interoperable, Reusable) 원칙에 따라 연구결과를 공유할 수 있음

· Figshare에 업로드된 항목은 최소 10년 이상 유지되며, Chronopolis 및 Duracloud를 통해 이중 보존됨

· 이용자(개인 혹은 기관)는 Figshare 웹사이트에서 계정을 만들고, 할당받은 용량 내에서 파일을 업로드하여 공유함

- Figshare에서 취급하는 데이터 유형 및 포맷은 다음 <표 12>, <그림 22>와 같음

<표 12> Figshare에서 저장할 수 있는 데이터 유형

데이터 유형	예시	비고
연구데이터	실험 데이터, 관측 데이터, 시뮬레이션 결과 데이터세트	정제된 데이터 또는 원시 데이터 모두 가능
출판물 관련 자료	논문 초고(preprint), 포스터, 컨퍼런스 발표자료	DOI 발급 가능, 공개 접근 설정 가능
소프트웨어 및 코드	Python 스크립트, R 코드, MATLAB 파일, 워크플로우	GitHub와 연동하거나 직접 업로드 가능
멀티미디어 파일	실험 사진(JPEG, PNG), 인터뷰 녹음(WAV), 실험 동영상(MP4)	1000개 이상의 파일 포맷 미리보기 지원
문서 및 텍스트 파일	PDF 논문, Word 문서, Excel 데이터, README 파일	데이터 설명용 문서 포함 권장
도구 및 연구 산출물	실험 프로토콜, 설문지, 인터뷰 질문지, 3D 모델 파일(CAD 등)	연구과정 기록용 문서 업로드 가능
기타	Null 결과 데이터, 부수 데이터(보조 실험 데이터 등)	미출판 데이터도 업로드 가능, 후속 버전 관리 가능

viewer type	thumbnail type	controls	extensions
archive	svg glyph		tgz, zip, rar, iso, tar, bz2, gz, xz, txz, 7z
audio	svg glyph		mp3, wav, aif, aiff, wma
dataset	svg glyph		csv, sav, tsv, xls, xlsb, xlsx
document	generated	zoom, pagination	doc, docx, dxf, odp, ods, odt, pages, pdf, rtf, ttf, xps, epub

<그림 22> Figshare에 업로드할 수 있는 데이터세트 포맷

- 메타데이터

· DataCite 스키마를 기반으로 하며, 해당 표준을 준수하고 있음. 모든 메타데이터는 관계형 데이터베이스에 저장되고, 필요에 따라 다른 메타데이터 스키마로 매핑 가능함

· API를 통해 아이템(Item), 컬렉션(Collection), 프로젝트(Project)에 대한 메타데이터를 JSON 형식으로 조회 가능함

· 메타데이터는 '건(Item)'단위로 작성하고, 표준은 Dublin Core, DataCite, National Library of Medicine, BibTeX, RefWorks, EndNote, RIS 형식을 제공함

· 기술 메타데이터는 총 30개로, 제목, 저자, 건 유형, 카테고리, 키워드, 기술, 라이선스 등으로 구성되어, 이용자에게 제공하고 있음

· <그림 23>은 시스템에서 자동으로 수집하는 기술 메타데이터로, 버전, 크기, DOI, 건의 URL, 생성일, 변경일 등이 있음

- 데이터 무결성 정책

· 모든 Figshare 사용자는 계정을 생성하고 항목 등록 또는 데이터 업로드를 위해 유효한 이메일 주소를 제공해야 함. 추가로, 사용자는 ORCID 프로필을 계정에 연동하여 신원을 보강할 수 있음

· Figshare는 파일과 메타데이터 모두에 대해 버전 관리를 지원함. 공개된 콘텐츠는 각각의 랜딩 페이지(landing page)에 이전 버전들이 표시되며 접근 가능함

· 파일이 Figshare에 업로드될 때 MD5 무결성 검사가 수행됨. 생성된 MD5 값은 레코드의 파일 미리보기와 함께 표시되고, 다른 메타데이터와 함께 저장되어 추후 무결성 검사를 위해 활용됨. 또한 Figshare의 기본 저장소인 Amazon AWS S3는 정기적이고 체계적인 데이터 무결성 검사를 실시함

Field	DataCite Field	Field Limits	API field name	Description	
Version	-	0:Configurable	version	DataCite maintains version information through the DOI	
Size	sizes	-	size	Size in bytes	
DOI	identifier	Valid format	doi		
Handle	-	Valid format	handle	Institution only	
Item URL	-	Valid format	figshare_url		
Created Date	-	yyyy-mm-ddThh:mm:ssZ	created_date	This is the date the version was published on Figshare.	
Modified Date	<dates><date> (dateType="Updated")	yyyy-mm-ddThh:mm:ssZ	modified_date		
Published Date	-	yyyy-mm-ddThh:mm:ssZ	published_date	This is the same as posted date and is used in searches when limiting by "published_since"	
Posted Date	<dates><date> (dateType="Created") and publicationYear	yyyy-mm-dd	timeline: posted	Overwritten by "Publisher date" if available	
Item Type ID	-	integer	defined_type		

<그림 23> Figshare의 수집 메타데이터

- 평가 및 분석

· Figshare는 관계형 데이터베이스를 직접적으로 보존하는 시스템이 아님. 주로 파일 형태의 연구 산출물을 저장하는 플랫폼임. SQL이나 SIARD와 같은 DB 덤프 파일을 저장할 수는 있지만, 그 내부에서 데이터를 구조적으로 꺼내는 기능은 없음

· 사진, 영상, 오디오, 코드, 멀티미디어 자료 등 대용량 바이너리 객체(BLOB) 파일 지원이 가능하며, 업로드된 자료는 미리보기가 가능함

· 데이터는 '건(Item)' 단위로 저장되며, 관계형 데이터베이스의 테이블 단위 추출은 지원하지 않음. 대신 개별 파일 단위로 버전 관리 및 접근 가능함

· CSV, JSON, PDF 등 표준 오픈 포맷을 지원하고, 저장 인프라는 Amazon S3 기반의 개방형 구조이며, 메타데이터 형식도 국제표준을 따름

○ **(과학기술)** NSF(미국 국립 과학 재단)의 DataONE

- 개요

· 지구 데이터 관측 네트워크(Data Observation Network for Earth, DataONE)는 NSF 지원 및 기타 연구 프로젝트를 통해 수집된 데이터의 장기 접근성과 재사용을 지원하기 위해 핵심 서비스와 인프라를 제공하는 시스템임

· DataONE은 노드(Node) 기반 네트워크를 구성하고 있으며, 회원 노드(Member Node, MN)와 조정 노드(Coordinating Node, CN)로 구분함

· 회원 노드는 데이터와 메타데이터를 보관하는 저장소로, 특정 연구 커뮤니티를 대상으로 서비스를 제공하며, DataONE이 정한 표준 인터페이스를 사용해 다른 노드들과 상호 운용함. 조정 노드는 이러한 회원 노드들이 하나의 연합체처럼 통합 운영될 수 있도록 네트워크 서비스를 제공함

· 각 노드에는 고유하고 변하지 않는 노드 식별자(Node Identifier)가 부여되며, 이는 시스템 안에서 해당 노드를 가리키는 기준이 됨. URL 같은 서비스 접속 주소는 시간이 지나 바뀔 수 있지만, 노드 식별자는 변하지 않아 지속적인 참조가 가능함

· DataONE의 모든 구성요소는 DataONE API를 통해 상호작용함. 회원 노드는 Member Node API, 중앙 관리 시스템인 조정 노드는 Coordinating Node API를 제공함. API는 RESTful 패턴을 따르며 HTTPS로 통신하고, 주고받는 메시지는 XML 형식으로 인코딩됨. 이 XML은 DataONE의 전용 XML 스키마로 정의되며, 오류가 발생하면 HTTP 오류 코드와 함께 XML 형식의 오류 정보가 전달됨

- DataONE의 데이터

· DataONE에서는 각 데이터와 함께 과학 메타데이터를 제공하는데, 이 메타데이터는 데이터의 속성과 특징을 설명하는 역할을 함. 또한 각 데이터나 과학 메타데이터에는 시스템 메타데이터(system metadata)라는 문서가 붙는데, 여기에 해시값, 타임스탬프, 소유권, 관계 정보 같은 디지털 객체의 속성이 기록됨

· 현재 DataONE은 실제 데이터를 바이트 집합으로 취급하며, 내용이나 형식에는 직접 개입하지 않음. 이런 데이터는 주로 회원 노드에 저장됨. 대신, 과학 메타데이터는 조정 노드에도 복사되어 저장되고, 검색을 위해 속성을 분석·추출함. 이렇게 하면 사용자가 메타데이터를 통해 원하는 데이터를 쉽게 찾을 수 있음

- 메타데이터

· DataONE가 활용하는 데이터세트 및 데이터베이스 관련 메타데이터 표준은 아래 <표 13>과 같음

· 그 외에도 'Darwin Core, ISO 19137, NEXML, Genbank, GM Profiles, MAGE, ESML(Earth Science Markup Language), MIENS 등이 있음

- 데이터 패키지

· DataONE의 데이터 패키지는 최소 한 개 이상의 데이터 파일과 한 개 이상의 메타데이터 문서로 이루어짐. 그리고 이 둘이 어떻게 연결되는지를 설명하는 리소스 맵(Resource Map)이라는 문서가 함께 포함됨

· 이 리소스 맵은 OAI-ORE라는 국제표준 형식으로 작성해야 하며, 이 표준은 데이터를 RDF 방식으로 표현함. 현재 DataONE에서는 여러 표현 방식 중 RDF/XML 형식만 지원하고 있음

· 또한 DataONE은 Java와 Python 환경 모두에서 리소스 맵을 만들고, 이를 저장 형식으로 변환하거나(직렬화), 다시 원래 구조로 복원하는(역직렬화) 기능을 제공함

- 데이터 보존 1단계

· 비트 보존 최우선: 데이터와 메타데이터의 비트 손상이나 소실을 방지해야 함

· 영구 식별자 부여: 모든 콘텐츠에 PID를 부여함

· 복제본 생성: 데이터세트는 총 3개(원본 1 + 복제 2)를 서로 다른 회원 노드에 분산 저장하고, 메타데이터는 4개 인스턴스를 유지함

· 회원 노드 유형: Read-Only, Replication-Open, Replication-Only로 구분함

· 복제 자동화: 데이터 등록 시 자동으로 복제하며, 복제 수는 회원 노드 상황과 정책에 따라 조정할 수 있음

· 주기적 검증: 무작위 샘플을 추출해 체크섬을 재계산하고 검증하며, 손상 시 복구하고 미디어 갱신(media refresh)을 권장함

<표 13> DataONE가 따르는 메타데이터 표준

표준명	설명
Dublin Core	리소스 설명을 위해 사용되는 15개 속성(properties)으로 구성된 메타데이터 어휘
GCMD DIF	NASA의 IDN 데이터베이스를 위한 메타데이터 "컨테이너" 역할을 함. 필수 항목, 키워드, 담당자 정보 등을 검증하며, 데이터 그룹을 설명하는 디렉터리 항목을 생성하는 데 사용. DIF는 사용자가 데이터세트의 유용성을 판단할 수 있도록 필요한 정보 제공
EML (Ecological Metadata Language)	EML은 생태학 분야를 위해 개발된 메타데이터 사양. Ecological Society of America의 연구결과를 기반으로 하며(XML 기반 문서 형태), 생태학 데이터의 다양한 측면을 모듈화하여 문서화할 수 있도록 설계됨
FGDC CSDGM	디지털 지리공간 메타데이터 콘텐츠 표준(Content Standard for Digital Geospatial Metadata, CSDGM)은 미국 연방 정부의 공식 메타데이터 표준. 1995년 이후 모든 연방 기관이 이 표준을 사용하여 지리공간 데이터를 문서화하도록 명령되었으며, 현재는 주 정부와 지방 정부에서도 광범위하게 채택됨
ISO 19115	지리정보(Geographic Information) 및 관련 서비스를 설명하는 국제 표준. 400개 이상의 메타데이터 요소를 정의하며, 핵심 요소 20개를 포함
Dryad Metadata Profile	Dublin Core를 기반으로 한 메타데이터 응용 프로파일. 시맨틱웹(Semantic Web) 개발과 호환되도록 설계됨
ADN	교육 환경(예: 교실 활동, 수업 계획, 시각자료 등)에서 활용되는 자원을 기술하기 위한 메타데이터 프레임워크
NetCDF-CF-OPeNDAP	대규모 과학 데이터(특히 기후 및 해양 데이터)의 접근 및 상호운용성을 지원하는 데이터 포맷 및 프로토콜
DDI (Data Documentation Initiative)	사회과학 데이터(예. 인구조사, 설문조사 데이터)의 기술 문서를 위한 국제 표준으로, XML로 작성함
WaterML	수문학 데이터를 교환하기 위한 정보 교환 스키마. 수자원 정보시스템(HIS) 프로젝트를 통해 개발되었으며, 다양한 수문학 데이터 제공자 간의 일관된 데이터 공유를 목표로 함
CSR (Cruise Summary Report)	해양 연구 프로그램에서 수집된 관측 및 샘플 정보를 문서화하기 위한 국제표준

- 데이터 보존 2단계

· 세 가지 보존 고려 요소: 디지털 콘텐츠를 보존할 때는 형태(Form), 의미(Meaning), 동작(Behavior)을 함께 고려해야 함. 형태는 콘텐츠의 구조적 표현, 의미는 그 안에 담긴 추상적 의미, 동작은 사용자가 의미를 이해할 수 있도록 제공하는 구체적 행위를 뜻함

· 권리 이해하기: 데이터를 보유할 권리, 보존 관리를 위해 복제·변환할 권리, 그리고 필요한 경우 후속 아카이브로 권리를 이전할 가능성을 확보해야 함

· 노후화 대응: 포맷이 오래되어 쓸 수 없게 될 때를 대비해 마이그레이션(Migration)과 에뮬레이션(Emulation) 전략을 사용함. 최근 가상화 기술로 에뮬레이션의 실현 가능성이 높아짐. 마이그레이션은 일부 회원 노드에서만 구축 가능하며, 모든 마이그레이션 과정은 버전 관리와

변환 전후 특성화를 거쳐야 함. 워크플로우가 복잡할수록 에뮬레이션 난도가 높아질 수 있음

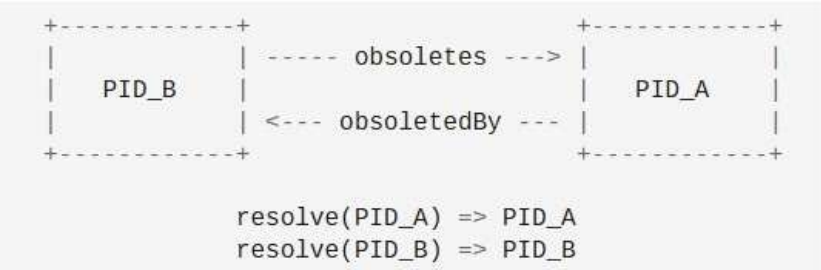
· 감시와 알림: 복제본, 원본, 외부 환경을 지속적으로 모니터링하며 AONS II, CRiB, PLATO와 같은 기존 알림 서비스를 활용함

- 콘텐츠의 변경 가능성(Mutability of Content)

· 변경 불가성: DataONE의 모든 콘텐츠는 수정할 수 없는 형태이며, 영속 식별자(PID)를 해석하면 항상 원본과 동일한 바이트 집합을 가리키는 URI를 얻게 됨

· 시리즈 식별자(SID): PID와 달리, SID는 항상 해당 객체의 최신 수정본을 가리키는 URI를 반환함

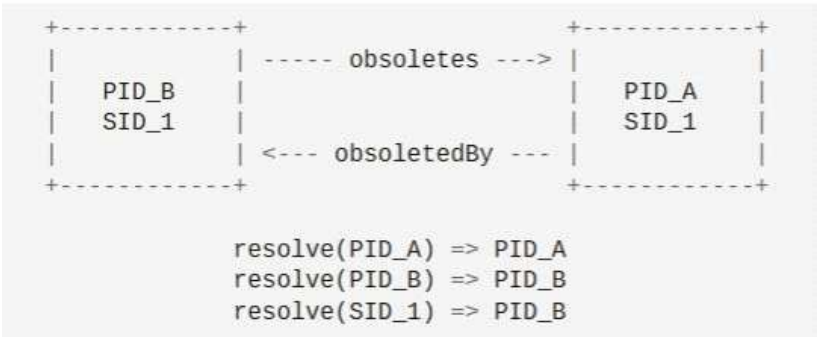
· 수정본·폐기 체인: 새 버전의 데이터가 나올 때, 이전 버전과의 연결 관계를 메타데이터에 기록하는 방식임. 예전 버전(PID_A)의 메타데이터에는 “obsoletedBy = PID_B”라고 적어서 “이 데이터는 PID_B에 의해 대체됐다”라고 표시함. 새 버전(PID_B)의 메타데이터에는 “obsoletes = PID_A”라고 적어서 “이 데이터는 PID_A를 대체했다”라고 표시함. 양쪽 메타데이터에 서로를 연결해 두어 버전 추적이 가능하게 하는 체계임(<그림 24> 참조)



<그림 24> 수정본·폐기 체인 예시

· 예전 DataONE 1 버전에서는 최신 버전을 찾으려면 사용자가 폐기 체인을 하나하나 따라가야 했음. 하지만 2 버전부터는 시리즈 식별자(SID)를 쓰면 이 과정이 훨씬 간단해짐

· 폐기 체인에 있는 모든 버전은 같은 SID를 공유함. 예를 들어 “SID_1”을 해석하면, 체인에서 가장 최신 버전(PID_B)의 URI를 바로 얻을 수 있음. 즉, PID는 특정 버전을 정확히 가리킬 때 사용하여 분석 결과 재현성 보장하는 데 유리하고, SID는 항상 최신 버전을 가리킬 때 사용하여 최신 데이터 기반 분석에 유용함(<그림 25> 참조)



<그림 25> DataONE 2 버전의 SID 예시

- 평가 및 분석

· DataONE은 다양한 분야의 데이터 형식을 저장할 수 있지만, 관계형 데이터베이스 자체를 구조적으로 그대로 보존하는 것은 어렵다는 한계가 있음

· DataONE은 데이터의 형식이나 구조를 변경하지 않고 원형 그대로 저장·전달함

· 테이블 단위 추출 기능은 없으며, 객체나 파일 단위로 관리됨. 그러나 메타데이터를 통해 세부항목 검색·필터링이 가능하고, RDF 리소스맵을 활용해 구성요소별 선택적 접근이 가능함

· 메타데이터 및 리소스맵은 RDF/XML, ISO, Dublin Core 등 국제표준을 기반으로 설계됨

· RESTful 방식과 XML 포맷을 사용하여 다양한 시스템과의 연계성을 확보함

· 데이터 자체는 어떤 파일 포맷도 저장 가능함

○ **(과학기술)** Bergen University(베르겐 대학교)의 ECKOchain

- 개요

· 생태학 데이터를 공개하고 장기보존을 보장하기 위해, 연구진은 블록체인 기술을 활용한 탈중앙화 데이터 관리 방식을 제안함

· 블록체인 기반 데이터베이스는 네트워크 구성원들이 합의한 투명하고 변경 불가능한 데이터 관리 프로토콜에 의해 운영됨. 특수화된 프로토콜은 새로운 데이터를 수용하기 전에 네트워크 전체의 합의를 보장하며, 어떤 개체도 기존 데이터를 단독으로 변경할 수 없음

· ECKOchain은 Hyperledger Fabric 프레임워크를 사용해 제작한 생태학 블록체인 기반 데이터베이스임

· 메타데이터와 접근 정책은 네트워크 모든 구성원에게 분산되지만, 기본 데이터는 데이터 소유자가 보유하며, 명시된 사용 라이선스에 따라 승인된 요청자에게 온디맨드 방식으로 제공됨. 데이터 요청의 세부사항은 블록체인에 영구적으로 기록되어, 감사 가능한 데이터 사용 계약으로 기능함

- ECKOchain

· ECKOchain은 Hyperledger Fabric이라는 오픈소스 기술로 만든 허가형(permissioned) 블록체인임. 즉, 개인이 아니라 조직 단위로 운영되고, 참여 조직만이 데이터 업로드와 다운로드를 할 수 있음

· ECKOweb은 블록체인에 접속하는 웹 포털로, 베르겐 대학교가 운영함. 검색과 목록 조회는 누구나 가능하지만, 실제 데이터 내려받기나 새 데이터 제출은 참여 조직 소속 사용자만 가능함. 각 조직은 ECKOweb 외에 자체 웹사이트를 만들어 연결할 수도 있음

· 데이터 제출 규칙은 데이터세트 자체를 표준화할 필요는 없지만, 메타데이터는 반드시 표준 형식을 따라야 함. 여기서는 Darwin Core(DwC)라는 국제표준을 메타데이터 포맷으로 사용함

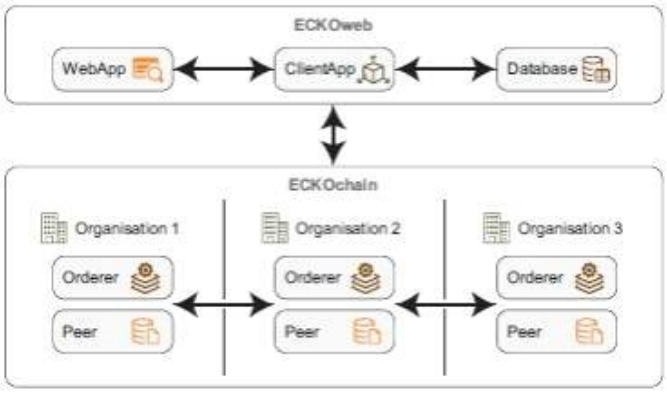
· 데이터 보안과 프라이버시 보호: 민감한 데이터(예. 멸종위기종 위치 정보)는 블록체인이 아니라, 데이터 소유 조직의 서버(오프체인)에 안전하게 보관됨. 사용자가 데이터를 요청하면, 설정된 라이선스와 접근 정책에 따라 온디맨드로 제공함

· 파일 무결성을 보장하기 위해, 제출 시 내용 기반의 고유 식별자(디지털 지문)를 생성해 블록체인에 기록함. 사용자는 파일을 받은 뒤 다시 디지털 지문을 생성해 블록체인값과 비교함으로써, 파일이 변조되지 않았음을 검증할 수 있음

- ECKOchain 구현

· ECKO는 블록체인(ECKOchain)과 웹사이트(ECKOweb)로 구성됨(<그림 26> 참조)

· ECKOchain은 참여하는 각 조직이 직접 운영하는 컴퓨터(피어 노드)가 모여 유지되는 블록체인임. ECKOweb은 사용자가 직접 블록체인에 접속하지 않아도, 대신 데이터를 읽고 쓰는 역할을 해주는 웹 포털임



<그림 26> ECKOchain의 구조

· ECKOweb은 웹사이트 속도를 높이기 위해 PostgreSQL 데이터베이스를 임시 저장소로 사용함. 블록체인 노드에 있는 데이터가 주기적으로 이 데이터베이스로 복제되며, 사용자가 웹사이트에서 데이터를 요청하면 블록체인 노드를 직접 조회하지 않고 데이터베이스에서 바로 가져오기 때문에 분산 인프라의 부담이 줄어듦

· 데이터 관리 규칙은 Java로 작성된 스마트 계약(체인코드) 안에 들어있음. 이 스마트 계약은 새로운 데이터가 제출될 때, 필수 메타데이터, 접근 정책, 기본 데이터 파일이 모두 갖춰져 있는지를 확인함

· 실제 데이터 파일은 데이터 소유 조직의 오프체인(private data collection)에 저장되고, 메타데이터와 접근 정책만 블록체인에 기록됨

· <그림 27>과 같이, 제공자(Contributors)가 데이터, 메타데이터, 정책을 제출하면, 스마트 계약이 이를 검증함. 메타데이터와 정책은 블록체인(Shared Ledger)에 기록되고, 실제 데이터는 조직의 개인 저장소(Private Data)에 보관함. 소비자(Consumers)가 요청할 시, 정책을 확인하고 데이터를 제공함

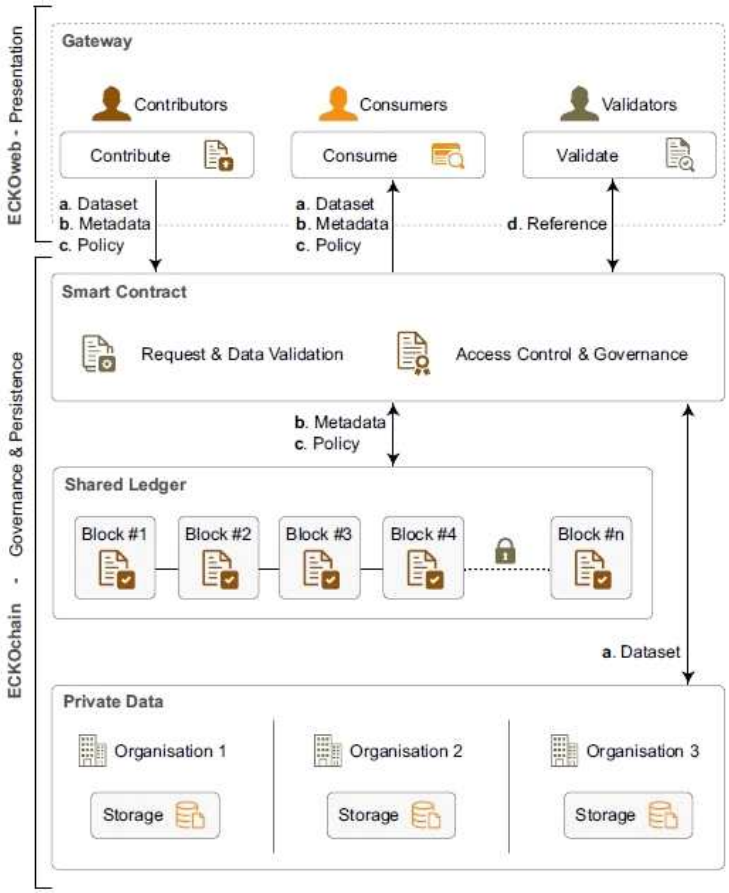
- 평가 및 분석

· 다양한 데이터베이스 수용 여부: 관계형·비관계형 데이터베이스를 그대로 저장하는 것은 어렵지만, 여러 형태의 데이터를 파일 단위로 저장할 수 있음

· 외부 첨부파일 포함 여부: 대용량 파일, 사진, 음성, 영상 등 모든 종류의 바이너리 데이터를 오프체인에 저장할 수 있으며, 디지털 지문(해시)를 통해 변조 여부를 검증함

· 일부 테이블만 추출 가능 여부: 데이터베이스의 테이블 단위 추출은 불가능하고, 파일 단위 접근만 가능함

· 파일포맷의 개방성 여부: 메타데이터는 Darwin Core라는 국제표준을 사용하며, 실제 저장되는 파일 형식은 제한 없이 허용함. 무결성 검증은 공개 해시 알고리즘을 사용함



<그림 27> ECKOchain의 전체적인 구조 및 흐름

○ **(과학기술)** Bahria University(바리아 대학교)의 DBPreserve

- 개요

· 행위에 대한 증거, 기관의 기억, 역사적 및 과학적 가치 등의 데이터는 장기보존이 요구되고, 디지털 데이터의 장기 접근성과 무결성을 보장하기 위한 디지털 보존(digital preservation)이 필수적임

· 메타데이터는 사용된 소프트웨어뿐 아니라, 데이터 생성에 관여한 인물과 그 이유까지 포함해야 함. 보존 시에는 디지털 객체가 원래 운영 환경에 지나치게 의존하지 않도록 하는 것이 중요함

· 본 연구는 관계형 데이터베이스에 적용 가능한 OAIS(Open Archival Information System) 참조모델을 따르고 있음. 데이터베이스와 관련 문맥 정보를 포함한 SIP(Submission Information Package)가 생성되고, 이를 바탕으로 디지털 보존팀이 AIP(Archival Information Package)를 생산하고 장기 보존함. 이 AIP는 관계형 모델을 다차원 모델(dimensional model)로 마이그레이션하는 방식을 사용하며, 실제 적용 가능한 변환 규칙도 제안하고 있음. 결과적으로, 생성된 AIP는 단순하고 플랫폼 독립적인 도구를 통해 향후 쉽게 접근할 수 있도록 설계됨

· 실질적인 장기보존 방식은 데이터베이스를 특정 DBMS에 종속되지 않도록 개방적이고 중립적인 포맷으로 변환하면서 의미 정보를 충분히 포함시키는 것임. 이러한 방향성은 소프트웨어 독립형 데이터베이스 보존 접근 방식에서 구체화되었고, 대표적인 예로 SIARD와 DBPreserve Suite가 있음

· SIARD: 스위스 연방 기록관은(SFA)는 관계형 데이터베이스의 장기보존을 위한 SIARD 방식을 제안함. 이를 위해 SIARD Suite라는 소프트웨어 도구를 개발하였으며, Oracle, Microsoft SQL Server, Microsoft Access, MySQL 등 다양한 관계형 데이터베이스를 SIARD 포맷으로 변환할 수 있도록 지원함

· DBPreserve Suite: DBPreserve Suite는 자바(Java)로 개발된 도구로, 다차원 모델링(dimensions and facts)에 필요한 메타데이터를 생성하는 데 목적이 있음. 이 도구는 스타 스키마(Star Schema) 구조의 사실 테이블(fact table), 차원 테이블(dimension table), 계층 및 수준(levels and hierarchies)에 대한 정보를 포함하는 메타데이터를 생성함

- 모델 마이그레이션 접근법

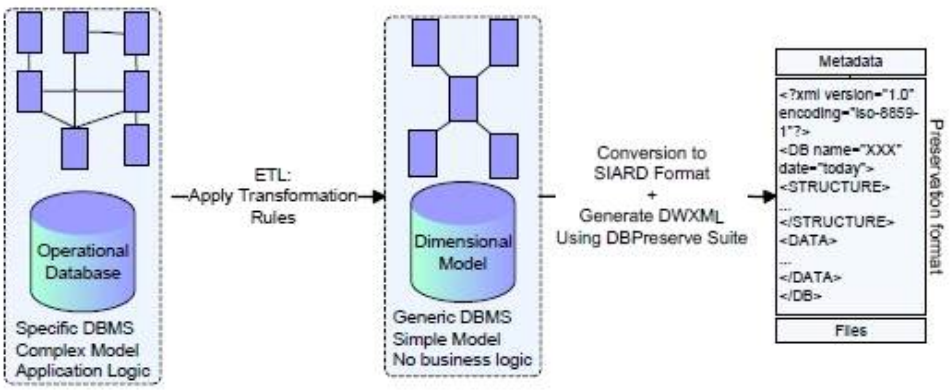
· 관계형 데이터베이스를 오래 보존하기 위해, 더 단순하고 직관적인 다차원 모델로 바꿔 저장하는 방법임(<그림 28> 참조)

· 관계형 데이터베이스는 구조가 복잡하고, 프로그램 안에만 들어 있는 규칙이나 지식이 많아 시간이 지나면 이해하고 쓰기 어려움. 운영이 끝난 데이터베이스는 계속 업데이트되지 않으니, 정규화된 복잡한 구조를 유지할 필요가 없음. 그래서 보기 쉽고 분석하기 좋은 형태로 바꿈

· 다차원 모델링이란 데이터를 차원 테이블과 사실 테이블로 나눠 저장함. 차원 테이블은 사람, 장소, 시간처럼 분석 기준이 되는 정보이고, 사실 테이블은 실제 수치나 사건 기록임

· 변환 절차(ETL 과정)는 기존 데이터베이스에서 데이터를 꺼내는 추출(Extract), 다차원 모델에 맞게 재구성하는 변환(Transform), 변환된 데이터를 새 모델에 넣는 적재(Load)로 구성됨

· 변환된 데이터는 XML 같은 독립적인 표준 포맷으로 저장해, 특정 DBMS 없이도 오래 쓸 수 있게 함. 이렇게 하면 데이터 내용뿐 아니라 그 안에 담긴 비즈니스 규칙과 의미까지 함께 보존할 수 있음



<그림 28> 데이터베이스 장기 보존을 위한 모델 전환 접근법

- 변환 규칙(Transformation Rules)

· 관계형 데이터베이스를 다차원 모델로 옮길 때 지켜야 하는 변환 규칙 중 첫 단계인 준비와 정리 단계임

· 그 목적은 기존 데이터베이스 구조를 분석하여 다차원 모델 설계에 활용하기 위함이고, ETL 과정에서 원본 데이터를 추적할 수 있게 유지하기 위함임

· 먼저, 테이블·컬럼 정보를 기록함. 테이블 이름, 의미, 생성일, 행 수, 컬럼 수, 제약조건, 대용량 객체(LOB) 포함 여부를 적음(<그림 29> 참조). 컬럼은 데이터 타입, 값 범위, NULL 값 개수 등을 기록하며, 의미는 사람이 직접 입력함(<그림 30> 참조).

· 값이 없는 컬럼이나 사용되지 않는 테이블은 제거함

Table Name	CONTRACTS
Meaning	This table contains data about the contracts of employees. Information like the start and end date, salary, type, duration, renewable or not about the contract are included in the table.
Source	Data is added to the table each time a new employee is hired or his contract is renewed. There is a form for adding a new employee in the application software.
Number of Rows	6220
Number of Columns	36
Date of Creation	
Date of Last Update	
Number of LOB Columns	0
Comments	This table contains data about all the contracts of an employee. The `personnel` table also contains data about the current contract of an employee which seems repetition and may be ignored.

<그림 29> 테이블 수준 기술 예시

Column Name	Meaning	Source	Data Type	Number of Nulls	Distinct Values	Minimum Value	Maximum Value
codpessoal	Reference to personnel table	New employee addition form	char	0	2224	AA	ZMAV
organization	The organization where the employee works	....	char	1256	10	0032	6
codcadre	Cadre Code	....	char	4	13	1	9
category	Contract Category	....	char	1657	260	0	930
invited	If the employee is invited from another institution	....	char	5666	9	0	V
inicfunccateg	Start date in a category	....	date	5924	175	1969.08.22	2008.12.31
constractsit	....	....	char	6023	12	A	XXX
organinicfunc	....	....	date	5673	27	0	99016
duration	....	....	number	268	40	0	51
durationtype	....	....	char	3219	2	A	M
renewal	....	....	char	4247	5		S
motiendcont	....	....	char	629	20	0	22
remarks	....	....	char	4036	1998		ú

<그림 30> 컬럼 수준 기술 예시

· 기본키 식별: 기본키는 테이블 병합 및 분할을 위한 유일 식별자이며, 다차원 모델에서 차원 테이블과 사실 테이블을 연결하는 데 필요함

· 찾는 방법은 (a) DB에 이미 제약조건으로 정의돼 있으면 그대로 사용, (b) 제약조건은 없지만, 딱 하나의 후보 키가 있으면 그것을 기본 키로 사용, (c) 후보 키가 여러 개면, 외래 키로 참조되는지를 보고 결정하거나 (d) 기본 키가 아예 없으면, 새로 '번호(ID)' 컬럼을 만들어 부여함

· 외래 키 식별: 모든 테이블을 분석하여 외래 키를 식별함. 명시적 제약조건이 없을 경우, 칼럼의 의미와 내용을 분석하여 참조 관계를 추론함. 외래 키 식별 시 고아 레코드(Orphan Child Records, OCR)가 존재할 수 있고, 다음 세 가지 방식으로 처리할 수 있음

· (a) OCR이 잘못된 데이터라면 삭제할 수 있지만, 무결성 훼손 여부를 반드시 확인하고, (b) OCR에 대응하는 부모 레코드를 새로 삽입함. 단, 이 경우 향후 사용자 오해의 소지가 없는지 검토해야 함. (c) 모든 OCR에 대해 "알 수 없는" 부모 레코드를 하나 삽입하고 이를 참조하게 함. 단, 이 경우 원래 값이 소실됨

- 정규화(Normalization)

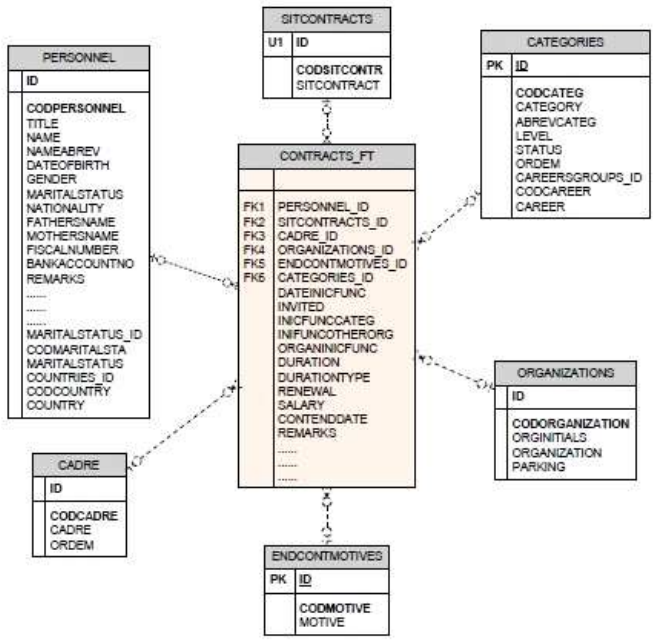
· 축약어 풀기 & 테이블 정규화: 데이터의 축약 코드(A=승인, N=미승인 등)는 풀어서 저장하며, 한 테이블에 혼합된 정보는 정규화해 분리하고, 다른 테이블에 유사 정보가 있을 경우, 중복 여부를 확인해 병합함

· 스타 스키마 만들기: 관련 있는 테이블끼리 묶어서 '클러스터'를 만들고, 그중 중심이 되는 테이블(가장 많은 데이터와 연결된 테이블)을 사실 테이블로 지정하여 주변에 있는 참고용 테이블은 차원 테이블로 지정함. 예를 들어, <그림 31>의 Contracts, Echelon, AccessRights, Trainings 테이블은 중심 테이블(사실 테이블)이고, Personnel과 같은 테이블은 자주 참조되나 중심 역할을 하지 않아 차원 테이블로 사용됨

Clusters \ Tables	Categories	CareerGroups	Countries	AccessRights	Personnel	MaritalStatus	Echelon	Contracts	Cadre	Trainings	Organizations	SitContract	EndContMotives
Contracts	X	X	X		X	X		X	X		X	X	X
Echelon			X		X	X	X						
Access Rights			X	X	X	X							
Trainings			X		X	X					X		

<그림 31> 테이블 클러스터링

· 최종적으로 구성된 모델은 네 개의 스타 구조로 이루어짐. <그림 32>는 Contracts 스타 구조 예시를 보여줌.



<그림 32> Contracts 스타 구조 예시

· 코드 분석: 기존 애플리케이션의 입력/조회/보고서 화면(Form)을 분석해 어떤 데이터가 쓰였는지 기록하고, 데이터 생성 규칙(함수, 저장 프로시저 등)을 실행해 결과를 데이터에 포함함. 각 코드가 무슨 역할을 하는지 설명문서를 작성함

· 메타데이터 기록 & 검증: 변환과정 전체에 걸쳐 참조·맥락·기술·출처 메타데이터를 수집하고, ETL 변환 매핑 정보와 데이터 출처를 문서화하며, 변환 완료 후 원본과 비교해 데이터가 빠진 게 없는지 확인함. 모든 차원과 사실 테이블이 완성되면 AIP로 만들어 장기보존함

· 보존 정보패키지 (AIP: Archival Information Package)

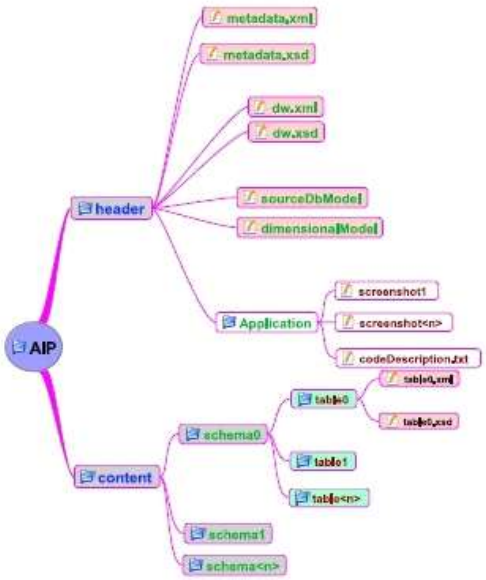
· 모델 마이그레이션된 데이터베이스는 SIARD 형식으로 변환되어 완전히 플랫폼 독립적인 형태로 포함됨. 이 데이터베이스에는 간단한 구조의 데이터 값뿐만 아니라 일부 맥락적 메타데이터도 함께 저장됨

· 변환된 데이터베이스는 SIARD 형식으로 저장하여 특정 프로그램·운영체제에 묶이지 않음. 또한 SIARD Suite를 통해 테이블·컬럼 설명, 담당자 정보, 소유자 정보 등의 메타데이터를 넣음

· DBPreserve Suite를 사용해 차원 모델 관련 메타데이터를 자동 생성하고 SIARD에 추가함

· 보존 로그(preservation log)는 데이터베이스 보존 절차 동안 수행된 모든 작업 기록을 담고 있

음. 이는 전환 규칙에 따라 수행된 단계들을 상세히 포함하며, 텍스트 파일 형식으로 저장됨



<그림 33> DBPreserve의 AIP 구조

- 데이터베이스 보급(Database Dissemination)

· 보존된 데이터베이스에 접근하는 방법은 두 가지가 있음

· SIARD Suite로 복원: 이 방법은 사용자가 SQL과 같은 질의 언어에 대한 지식을 갖추고 있어야 하며, 또한 DBMS가 설치된 컴퓨터 환경도 필요함

· DBPreserve Archive Browser(DAB)로 바로 보기: 자바(Java)로 만든 뷰어로 DBMS 없이 SIARD 파일을 열람할 수 있음. 원하는 테이블을 선택하면, 관련된 다른 테이블 목록을 자동으로 표시해 줌. 주요 기능은 테이블 필터링(특정 단어로 테이블 전체나 특정 열에서 검색), 키 따라가기(외래 키 클릭 → 연결된 부모 데이터 보기 / 기본 키 클릭 → 참조하는 데이터 보기), 테이블 조인(기준 테이블과 연결된 다른 테이블을 합쳐서 보기)가 있음

- 평가 및 분석

· 다양한 데이터베이스 수용 여부: DBPreserve는 SIARD 포맷과 함께 Oracle, SQL Server, MySQL, Access 등 주요 DBMS를 지원하며, 다양한 스키마 구조에 적용 가능함. 관계형 데이터베이스를 다차원 구조로 추상화하여 저장하므로 DBMS에 종속되지 않고 상호운용성을 확보함

· 외부 첨부파일 포함 여부: 주로 정형 관계형 데이터 보존에 초점을 두며, 첨부파일이나 외부 참조 데이터 포함에 대한 명시적 서술은 없음

· 일부 테이블만 추출 가능 여부: 전체 데이터베이스 또는 클러스터 단위로 테이블을 분류하고, 버스 매트릭스 및 클러스터링을 통해 필요한 테이블만 선택적으로 보존할 수 있음. 분석 목적에 따라 일부 테이블만 추출하여 활용 가능함

· 파일포맷의 개방성 여부: 국제표준 오픈 포맷인 SIARD를 사용하며, Java 기반의 오픈소스 도구(DBPreserve Suite, DAB)를 통해 구현됨. XML 기반 구조로 설계되어 있어 다양한 소프트웨어 환경에서 장기 활용 가능성이 높음

○ **(과학기술)** Durham University의 HEPData

- 개요

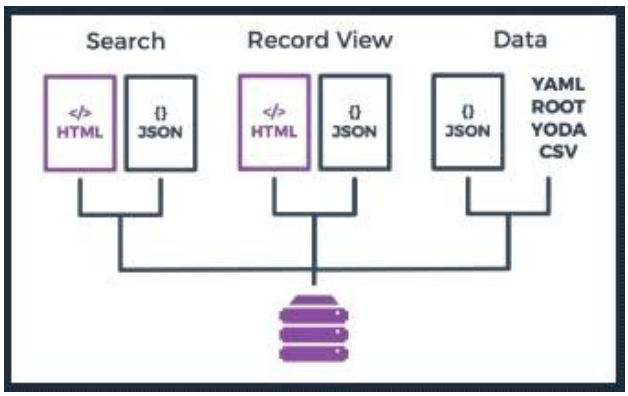
· 고에너지물리학(HEP) 분야에서는 실험 종료 이후에도 데이터가 과학적·사회적 가치를 지속적으로 창출할 수 있도록 장기적 활용 가능성을 보장하는 체계가 필수적임

· DPHEP은 이러한 문제의식 속에서 2013년에 출범한 국제 협력체로, 단순한 비트 보존을 넘어 분석 재현과 재사용을 가능하게 하는 '실행 가능한 보존(executable preservation)'을 핵심 목표로 삼아 전략과 도구를 정교화해 왔음

· DPHEP이 제시하는 데이터의 장기보존, 접근성 강화, 재사용 촉진, 재현성 확보라는 목표를 바탕으로 HEPData 프로젝트를 실행함

· HEPData는 2014년에 개설되어 고에너지물리학 데이터를 관리하고 공개하는 플랫폼으로, 2.8 페타바이트 이상, 120만 개 이상의 파일을 보유하고 있음

· HEPData의 대부분 페이지는 HTML 보기 외에도 JSON 포맷을 제공함(<그림 34> 참조)



<그림 34> HEPData의 출력 형식

· HEPData 지원 포맷 및 지정 쿼리는 다음 <표 14>와 같음

- 평가 및 분석

· HEPData는 특정한 관계형 데이터베이스 덤프를 그대로 저장하는 시스템은 아님. 대신 실험에서 생성된 시뮬레이션 결과, 그래프 데이터 등 연구 산출 데이터를 다양한 파일 형식으로 제공함. 원본 데이터베이스의 종류와 상관없이 등록할 수 있어 데이터 출처의 제약을 받지 않는 개방형 저장 구조를 가짐

· CSV, YAML, JSON은 범용 오픈 포맷이며, ROOT와 YODA는 고에너지물리(HEP) 분야에서 널리 사용되는 공개 표준 전용 포맷임. 범용 포맷과 전용 표준을 병행 지원함으로써 다양한 분야의 데이터 활용 및 공유 가능성을 높임

· 그래프나 구조화된 수치 데이터를 ROOT 파일로 보존할 수 있으며, YAML with resource 방식을 통해 외부 리소스(이미지, 추가 데이터 등)를 함께 저장할 수 있음. 이를 통해 대용량 이진 데이터나 외부 첨부파일도 간접적으로 포함할 수 있는 유연한 보존 구조를 제공함

· table 옵션을 활용하면 특정 테이블만 선택적으로 다운로드할 수 있어, 데이터 활용 시 효율성과 접근성을 높일 수 있음

<표 14> HEPData 포맷 및 쿼리

구분	내용
지원 포맷	· 리소스 파일 포함 YAML: 원본 데이터 + 업로드된 추가 리소스 포함 · YAML: 리소스 파일 없이 DOI 정보 추가 · CSV: 쉼표 구분 텍스트 포맷(엑셀 등에서 사용 가능) · ROOT: .root 바이너리 파일, 수치 데이터는 TGraphAsymmErrors 등 ROOT 객체로 저장 · YODA: Rivet 툴킷에서 사용하는 분석 데이터 포맷 (YODA2 기본, 필요시 YODA1 선택 가능)
포맷 지정 쿼리	· ?format=json · ?format=original · ?format=yaml · ?format=csv · ?format=root · ?format=yoda · ?format=yoda1
추가 옵션	· table=Table%201: 특정 테이블만 다운로드 (공백은 %20으로 인코딩) · version=1: 특정 버전 지정 · light=true: JSON 경량 버전(데이터 테이블 제외) · rivet=ALICE_2016_I1419244: YODA 파일 생성 시 Rivet 분석 이름 명시

○ **(과학기술)** CERN(유럽 입자 물리 연구소)의 CERN 오픈 데이터 포털

- 개요

· 2014년에 개설되어 고에너지물리학 데이터를 관리하고 공개하는 플랫폼으로, 2.8페타바이트 이상, 120만 개 이상의 파일을 보유하고 있음

· 공개된 데이터는 맞춤형 메타데이터 스키마에 따라 서지 레코드 형태로 기술됨. 포털은 제목, 저자, 키워드, 연도와 같은 기본 메타데이터뿐만 아니라, 파일 포맷, 파일 크기, 파일 위치, 체크섬과 같은 기술 메타데이터(descriptive metadata), 그리고 데이터의 활용 방법, 데이터 의미를 설명하는 고수준 메타데이터도 함께 제공함.

· 각 서지 레코드는 데이터 인용과 참조를 용이하게 하기 위해 DOI(디지털 객체 식별자)가 부여됨

· 데이터는 EOS 저장 시스템을 통해 이용할 수 있고, CERN CTA(테이프 아카이브 시스템)에 백업되어 있음

· EOS 저장 시스템은 여러 서버와 디스크를 묶어 하나의 대규모 가상 스토리지 풀처럼 작동함. 데이터는 복제를 통해 여러 위치에 저장되어, 하나의 디스크나 서버가 고장이 나도 데이터가 안전하게 유지됨

· CERN은 약 18개월마다 데이터를 새로운 저장 장치(테이프)로 복사하여, 모든 파일이 여전히 사용 가능한지 점검함

- 평가 및 분석

· CERN Open Data는 단순히 관계형 데이터베이스에서 추출한 표 형태 데이터만을 제공하는 것이 아니라, 분석에 필요한 코드와 실행 환경 설정까지 함께 묶어 제공함. 이는 데이터 그 자체뿐 아니라 재현을 위한 전체 분석 환경을 하나의 단위로 패키징하는 구조임. 관계형 DB 전체 덤프보다는 실험 데이터 분석에 적합한 전용 파일 형식 중심으로 운영되는 것이 특징임

· 데이터 패키지에는 분석 스크립트, 실행 환경 설명 파일, PDF 파일 등 다양한 형식의 자료가

함께 포함되며, 파일 경로를 참조하는 방식이 아니라 필요한 파일을 모두 하나의 패키지 안에 직접 포함하는 구조를 채택함. 이를 통해 장기보존 시점에서 실행 환경을 복원하거나 분석 절차를 재현할 수 있는 신뢰성을 확보함

· 전체 실험 데이터 중 특정 분석 단위만을 독립된 DOI와 함께 제공할 수 있으며, JSON 및 ROOT 내 테이블 구조를 기준으로 선별적 제공이 가능함

3.2 데이터세트(DB형) 유형의 기록물에서의 원문에 대한 이해

가.. 디지털화(Digitization) 또는 변환(Migration)에 따른 원문 폐기 근거 해외 법령 및 표준

○ 비전자기록물 매체 디지털(Digitization)에 따른 원문 폐기 근거 해외 법령 및 표준

- 미국 NARA(National Archives and Records Administration)

· 36 CFR Part 1236 – Electronic Records Management Subpart D, E

◦디지털화 대상으로 영구 종이 기록물과 사진 인쇄물을 포함하며 대상 기록물의 디지털화를 위한 절차와 품질 기준을 규정하고 모든 대상 기록물에 대해 디지털화 프로젝트의 범위, 품질 관리, 완전성 검증 등을 명시함

◦해상도, 색상 정확도, 비손실 압축 방식(TIFF, JPEG2000), 파일 포맷 요구사항 등의 품질 및 기술 기준을 구체적으로 규정함

◦디지털화된 기록물이 기록의 출처, 날짜, 디지털화 변환 절차, 담당자 식별 등 핵심 메타데이터를 포함하도록 요구함

◦기술 요건을 충족한 디지털화된 기록물이 검증 완료 후 조건을 충족하였을 때 원본기록물을 대체할 수 있고, 이때 기관은 원본기록물을 폐기할 수 있음

· GRS 4.5, Digitizing Records

◦디지털화가 완료된 원본기록물의 폐기를 위한 일반 권한을 규정한 문서로, 디지털화 기준을 충족한 경우 원본기록물은 폐기가 가능함을 명시함

◦GRS 4.5는 Subpart D와 E의 기준을 충족한 기록물 혹은 1950년 이후 생산된 기록물에 적용 가능함

◦디지털화가 완료된 원본기록물 중 폐기 대상이 아닌 기록물로는 Subpart D와 E의 기준을 충족하지 못한 기록물, 1950년 이전 생산된 기록물, 높은 가치를 지닌 기록물, 기술적인 이유로 인해 디지털화를 진행할 수 없는 기록물이 있음

· AC 31.2023 – Release of Regulations with Digitization Standards for Permanent Records

◦NARA는 메모 유형인 AC 31.2023을 통해 36 CFR Part 1236 규정을 발표하고 시행 일정(2023년 6월 5일 발효)을 공식적으로 안내함

◦기관이 Subpart D, E 기준을 준수할 경우 원본기록물을 폐기할 수 있음을 명시함. 다만, 원본기록물의 폐기는 유효한 기록물 처리 일정을 보유하고 있어야 가능함

◦AC 31.2023은 Subpart D, E 규정과 GRS 4.5가 동시에 적용됨을 명시함

· 다음 <표 15>는 NARA의 비전자기록물 디지털화에 따른 원본기록물 폐기의 근거가 되는 법률 및 표준을 정리한 것임. 36 CFR Part 1236은 디지털화의 절차적, 기술적 요건을 명시하고 있으며, 이는 GRS 4.5의 원본기록물 폐기의 요건을 충족시키기 위한 전제 조건으로 활용됨

<표 15> 미국 NARA의 디지털화 원문 폐기 근거

법률 / 표준	내용
36 CFR Part 1236 Subpart D, E	· 원본기록물 폐기를 위한 절차적, 기술적 요건 정의 · 디지털화된 기록물의 이미지 기술 사양 정의(해상도, 색상 정보, 압축 조건, 파일포맷 등) · 디지털화된 기록물의 지적 통제 및 물리적 통제를 위한 메타데이터 필수 요소 정의 · 디지털화 과정을 기록하는 별도의 전자기록물 생산 명시 · 디지털화된 기록물과 원본기록물 처분 권한에 대한 검증(Validation) 요건 정의
GRS 4.5 Digitizing Records	· 폐기의 대상이 디지털화된 기록물의 원본기록물임을 명시 · 디지털화된 기록물이 Subpart D, E의 기준을 충족할 때만 폐기 가능 · 디지털화된 기록물이 기준에 부합하는지 검증(Validation) 후 폐기 진행
	· 디지털화 과정의 관리 및 디지털화 결과물의 검증 결과를 기록한 문서를 보존해야 함 · 디지털화된 기록물 이관 완료 시 해당 기록물은 폐기 진행
AC 31.2023	· 36 CFR Part 1236과 GRS 4.5의 시행을 공식적으로 안내 · 디지털화된 기록물의 원본기록물 폐기에 관한 규정인 GRS4.5가 모든 연방 행정기관에 적용됨을 명시

- 호주 빅토리아주 PROV(Public Record Office Victoria)

· PROS 25/02 S1 Digitisation Specification

◦ 호주 빅토리아주의 공공기관이 기록물 디지털화를 진행할 때 따라야 하는 기술적·절차적 기준을 규정한 표준으로, 디지털화된 기록물의 이미지 품질, 메타데이터 요구사항, 검증 절차, 디지털화 과정의 문서화 등 디지털화 과정에 대한 전반적인 요구사항을 정의함

◦ 디지털화된 기록물이 원본기록물을 대체할 수 있는 조건이 명시되어 있으며 이를 기반으로 원본기록물을 폐기할 수 있는 요건을 명시

· PROS 19/07 Retention and Disposal Authority for Converted or Digitised Records

◦ 디지털화 또는 매체 변환이 이루어진 기록물에 대해 원본 폐기를 승인하는 기록물 보존 및 폐기 권한(Retention and Disposal Authority, 이하 RDA)에 대한 공식 표준

◦ PROS 19/07 RDA는 원본기록물 폐기를 위한 기술적(technical) 요건과 절차적 요건 등을 명시함. 이러한 요건을 모두 충족하면 디지털화된 기록물이 원본기록물을 대체할 수 있고 원본기록물 폐기가 가능함

· PROV 'Digitisation'

◦ PROV는 공식 홈페이지(https://prov.vic.gov.au/recordkeeping-government/a-z-topics/digitisation)를 통해 공공기관이 기록물을 디지털화하는 데 필요한 정책, 절차, 관련 문서 등을 안내하고 있음

◦ 원본기록물의 디지털화 절차, 디지털화된 기록물의 원본기록물 대체 조건, 디지털화 계획 수립 방법 등을 간략하게 서술함

◦ 원본기록물의 폐기에 관한 결정은 PROS 25/02 S1 표준과 PROS 19/07 RDA를 기준으로 하며, 디지털화 품질 검증과 폐기 판단에 대한 책임은 각 기관에 있음을 명시함

· 다음 <표 16>은 PROV의 비전자기록물의 디지털화에 따른 원본기록물 폐기의 근거가 되는 표준 및 관련 사항을 정리한 것임. PROS 25/02 S1은 기관이 디지털화 진행 시 만족해야 하는 요구사항을 정의하고 있으며 이는 PROS 19/07 RDA의 원본기록물 폐기를 위한 요건으로 활용됨

<표 16> 호주 PROV의 디지털화 원문 폐기 근거

표준	내용
PROS 25/02 S1 Digitisation Specification	· 모든 기록물은 RDA의 적용을 받고, RDA의 기준에 따라 원본기록물의 폐기 여부 결정 · 원본기록물이 완전하고 정확하게 디지털화되어 더 이상 원본 보존의 필요성이 없을 때 폐기 가능 · 원본기록물 폐기를 위한 디지털화된 기록물의 검증은 정확성과 완전성 및 법적·업무적 유효성을 충족해야 함 · 디지털화된 기록물의 원본 대체에 대한 책임은 기록물을 생산한 기관에 있음
PROS 19/07 RDA for Converted or Digitised Records	· 디지털화를 진행한 원본기록물의 폐기를 공식적으로 허용 · 원본기록물의 폐기 이전에 디지털화된 기록물이 완전하고, 정확하며, 접근 가능한지 확인해야 함 · 디지털화된 기록물은 기록관리 시스템에 등록되거나 적절한 메타데이터와 함께 유지되어야 함
PROV 'Digitisation'	· 원본기록물의 디지털화 진행 시 참고해야 할 기술적인 기준은 PROS 25/02 S1을 만족해야 함 · 원본기록물이 PROS 19/07 RDA에 따라 더 이상 보존할 필요가 없을 경우, 폐기 진행

○ 전자기록물 매체 변환(Migration)에 따른 원문 폐기 관련 사항

- 미국 NARA(National Archives and Records Administration)

· NARA Federal Electronic Records Modernization Initiative(FERMI) - Use Cases for Transfer

◦ FERMI는 효율적이고 표준화된 기록관리 솔루션 및 서비스를 제공하기 위해 개발되었으며 그 중 'Use Cases for Transfer'는 전자기록물의 이관 절차에 대한 시나리오 지침서로 사용됨

◦ 기관이 영구기록물인 전자기록물을 NARA에 이관한 후, 이관이 완료된 전자기록물의 사본에 대한 관리 절차가 명시되어 있음. 이 절차에서 '이관된 기록의 사본(copies of transferred records)'이라는 용어를 확인할 수 있음

· NARA Federal Electronic Records Modernization Initiative(FERMI) - Use Cases for Disposal

◦ FERMI는 효율적이고 표준화된 기록관리 솔루션 제공을 위해 개발되었으며 그 중 'Use Cases for Disposal'는 전자기록물의 폐기 절차 안내를 위한 시나리오 지침서로 사용됨

◦ 전자기록물의 폐기 업무는 보존기간이 만료된 전자기록물을 대상으로 하며 폐기 승인 절차와 폐기 실행 절차를 설명함. 폐기 대상의 기록물 유형으로는 클라우드 서비스, 구조화된 데이터(DB, XML 등), 이미지, 오디오, 비디오, 웹사이트 등이 있음

· <표 17>은 NARA의 전자기록물 이관 및 폐기와 관련된 사항을 정리한 것으로 관련 지침서에는 전자기록물 사본의 폐기에 관한 사항이 명시되어 있음

<표 17> 미국 NARA의 매체 변환 원문 폐기 관련 사항

표준	내용
FERMI Use Case for Transfer	· 기관에서 NARA로 전자기록물을 이관하였을 때, 해당 전자기록물에 대한 관리 책임은 기관에서 벗어나 NARA에 있음을 명시함 · 이는 기관이 보유한 전자기록물은 이관된 기록물의 사본이 되었음을 의미함 · 기관은 NARA로 이관된 전자기록물에 한해 사본 폐기가 가능함
FERMI Use Case for Disposal	· 폐기를 '보존 기간이 만료되고, 기관에 더 이상 업무적 가치가 없는 기록을 파기하는 행위'라고 정의함 · 기관은 폐기 시점에 모든 전자기록물의 사본의 폐기를 보장해야 함

- 호주 빅토리아주 PROV(Public Record Office Victoria)

· PROS 19/07 Retention and Disposal Authority for Converted or Digitised Records

◦ PROS 19/07 RDA에서 정의한 '변환(conversion)'은 디지털 기록에서 디지털 기록으로 변환하는 것을 포함

◦ 2000년 이후 생산된 모든 원본기록물은 별도의 조건을 충족하였을 때 폐기할 수 있으며, 2000년 이전의 원본기록물은 PROV의 별도 승인 이후 폐기 가능

◦ 폐기를 위해 만족해야 하는 조건으로는 변환이 완료된 기록물의 진본성, 무결성, 사용성 확보, PROV 표준과 최소 보존기간 준수, 원본 생성일 메타데이터 보존 등이 있음

· PROS 22/04 Disposal Standard

◦ 마이그레이션(Migration)을 비롯한 변환이 폐기를 의미하지 않으며, 원본기록물의 폐기는 별도의 요건을 충족하여 진행해야 한다고 명시함

◦ 기관은 폐기를 승인할 수 있도록 적절한 폐기 승인 수단을 보유해야 하며, 기록물을 폐기하기 위해서는 폐기 요건을 충족하고 승인된 폐기 수단을 사용해야 함

· PROS 10/13 Disposal Guideline - Implementing a Disposal Programme

◦ PROS 10/13 Disposal Guideline은 전자기록물의 분류 및 폐기에 관한 표준으로, 기록물 폐기 프로그램을 구현하는 가이드라인을 제시함

◦ 타 기관에서 이관받은 전자기록물은 체계적으로 검토 및 처리해야 하며, 시스템 간 호환성과 기록 식별 문제를 해결했을 때 전자기록물의 폐기가 가능

◦ 전자기록관리시스템(EDRMS) 내에서 기록물과 RDA를 연결하여 폐기 절차를 간소화할 수 있으며, 폐기 트리거(disposal triggers), 자동화된 워크플로우 등을 설정하여 시스템이 폐기 과정을 관리하도록 함

· <표 18>은 PROV의 전자기록물 이관 및 폐기에 관한 사항으로 PROV의 매체 변환 및 폐기 관련 표준에 '변환(conversion)'에 관한 정의와 원본 폐기를 위한 조건이 명시되어 있음

<표 18> 호주 PROV의 매체 변환 원문 폐기 관련 사항

표준	내용
PROS 19/07 RDA for Converted or Digitised Records	· 전자기록물의 매체 변환 시 원본기록물 폐기에 대해 공식적으로 명시함 · 변환된 전자기록물이 원본기록물의 목적을 충족하는 수준의 변환이 이루어져야 함 · 2000년 1월 1일 이후 생산된 기록물은 매체 변환 조건을 만족하였다면 PROV의 별도 승인 없이 폐기 가능함
PROS 22/04 Disposal Standard	· Migration은 폐기로 간주하지 않으며 기록물의 폐기를 위해서는 별도의 절차를 거쳐야 함 · 폐기 전까지 전자기록물의 접근성과 무결성을 확보한 상태로 보존해야 함
PROS 10/13 Disposal Guideline	· 기록물은 승인된 폐기 혹은 승인된 이관을 통해 폐기될 수 있음 · 폐기 활동의 자동화를 위한 가이드라인을 제시하고 있으며 자동화 프로그램을 위한 계획과 테스트에 대한 사항을 명시함 · 다른 기관에서 이관받은 전자기록물은 폐기 전 체계적인 검토가 필요하며, 시스템 호환성을 비롯한 문제들을 해결해야 함

나. 호주 빅토리아주 PROV(Public Record Office Victoria)의 VEO3에서 제시하는 원문 포함 방식과 서면으로 알아본 데이터세트형 기록물의 관리 사례

○ 호주 빅토리아주 표준: (PROV) PROS 19/05 S3: LONG TERM SUSTAINABLE FORMATS

- 아래 내용은 기록물을 이관할 때 장기보존용 파일포맷(보존포맷)으로 변환한 경우, 기존 원문을 VEO3에 포함해야 하는지에 대한 원칙을 설명한 절임. 본 표준에서는 원본 형식의 사본을 VEO3에 포함하는 것을 기본 원칙으로 명시하고 있음

- 다만 기록물의 크기가 매우 크고, 변환(Migration) 과정이 내용 손실 가능성이 거의 없는 일상적 기술 절차인 경우에는 PROV와의 협의를 통해 원문 사본을 VEO에 포함하지 않을 수도 있음을 설명하고 있음

2.5 Inclusion of Original Format
Where record content has been migrated to a long term preservation format for the purposes of transfer, a copy of the original, un-migrated format must also be included in the VEO unless otherwise agreed by PROV.

The requirement to include a copy of the original format guards against the following risks that:

- the migration did not result in an accurate representation of the original record
- the migration caused a loss of information
- a better migration approach may be available in the future.

PROV will not generally require a copy in original format where:

- *the record is extremely large, and*
- *the migration process is a routine technology process with little chance of content loss.*

A typical example of a situation where the original would not normally be required is the conversion of video or audio to a long term preservation format.

다. 디지털화 기록물 및 데이터세트형 기록물의 보존 방법에 관한 호주 빅토리아주 PROV 서면 질의 사항

○ 서면 질의서를 NARA, PROV에 송부했으나, PROV로부터 답변이 왔음

- 기록물의 매체 변환 이후 원본기록물의 보존·폐기 기준을 검토하기 위해 PROV에 서면 질의를 실시함. 이를 통해 데이터세트형 기록물의 보존체계를 분석하기 위한 기초 자료를 확보함

<표 19> PROV 매체 변환 및 데이터세트형 기록물 관련 서면 질의 및 답변

구분	내용	
질의1	질의	· VEO3에서 원본을 제외하여 패키징을 진행한 사례나 기준이 있는지 · 참고 표준: PROS 19/05 S3
	답변	· 실제 원본이 제외된 사례는 없지만, 향후 대용량 비압축 시청각 기록물에 대해 보존포맷으로 변환된 기록물만 VEO에 포함하고 원본은 별도로 보관하는 방안을 검토중 · 현재 이와 관련된 내부 기준은 별도로 마련되어 있지 않으며, 필요시 사례별로 판단할 가능성이 있음 · 용량이 매우 큰 파일 외에는 원본 폐기 사례는 없었으며, 두 가지 포맷을 제출한 기관의 경우 동일한 내용이지만 화질이 더 좋은 포맷을 선택한 사례가 있었음
질의2	질의	· 디지털화 이후 원본기록물 폐기 사례 및 공식기록물 지위 이전 사례가 있는지 · 참고 표준: PROS 25/02 S1
	답변	· 2000년 이전에 생산된 영구보존 기록물의 폐기는 PROV의 허가가 필요하며, 연 2건 내외의 폐기 허가 요청이 접수되었고 모두 승인된 전례가 있음 · 2000년 이후에 생산된 기록물은 허가 없이 폐기가 가능하며 폐기 보고 의무가 없으므로 실제 폐기 여부는 PROV에서 확인이 불가함 · 이에 따라, 디지털화 기록물이 공식기록물로 전환된 명시적인 사례를 확인하기 어려움
질의3	질의	· 데이터베이스와 연계된 애플리케이션 및 사용자 인터페이스 보존 사례나 전략이 있는지 · 참고 표준: PROS 19/05 S3
	답변	· 동적 컨텐트 구현용 애플리케이션을 보존한 사례는 없으며, 인터페이스와 애플리케이션을 보존하는 것과 에뮬레이션 전략은 현재 자원 한계로 비현실적이라고 판단됨 · SIARD와 같은 보존포맷을 통해 데이터 접근성을 확보하고, 기록시스템·애플리케이션의 구조와 기능을 기술하는 서술형 메타데이터로 보완함 · 데이터베이스 유형의 기록물은 이관 사례가 드묾

라. 데이터세트(DB형) 유형의 기록물의 원문에 대한 이해와 제안

❍ 비전자기록물 매체 디지털(Digitization)에 따른 비전자기록물(원문) 대체에 대한 시사점

- 미국(NARA)과 호주(PROV) 사례에서는 비전자기록물의 원문을 전자기록물로 대체하기 위해서는 기술적 기준 + 검증 절차 + 법적 승인이라는 삼중 구조가 필수적임을 보여주고 있음. 특히 디지털화의 “완전성 검증”과 “절차적 승인”은 원문 폐기의 핵심 요건이며, 이는 한국에서도 디지털화 기록물의 법적 신뢰성을 확보하기 위한 필수 조건으로 적용할 수 있음

❍ 전자기록물 매체 변환(Migration)에 따른 원문 대체에 대한 시사점

- 미국 NARA와 호주 PROV 사례는 공통적으로, 변환(Migration)된 전자기록물이 원본을 대체하려면 기술적·절차적·법적 요건을 모두 충족해야 한다는 점을 시사함. 특히 변환 과정의 검증(Validation)과 폐기 승인 절차는 원본 폐기 여부를 결정하는 핵심 요소로 작용함

❍ 호주 빅토리아 PROV의 VEO3의 원본 제외 요건과 전자기록물 장기보존 체계에 대한 시사점

- PROV는 VEO3에서 원본 포함을 기본 원칙으로 규정하지만, 대용량 기록물이면서 손실 위험이 거의 없는 일상적 변환이라는 두 조건을 동시에 충족할 경우에 한해 원본 제외를 허용함. 이 예외 기준은 장기보존에서 기술적 제약·저장 효율·관리 현실성을 고려한 유연한 운영의 필요성을 보여주며, 특히 데이터세트형 기록물 관리에서도 향후 원본 보존의 선택적 적용 가능성을 열어둠

- 또한, 데이터세트형(DB형) 유형 전자기록물에 대한 이관 사례는 드물고, 초보적인 단계로 예측됨

○ 국내 데이터세트(DB형) 유형 전자기록물의 기록관리 및 이관 관점에서의 원문에 대한 시사점

- **(전자기록물 원문의 전통적 개념)** 과거 전자기록물은 종이기록물을 대체하기 위해 단일 전자문서 형태로 생산되었기 때문에 대부분 하나의 파일포맷으로 표현할 수 있었음. 오디오·비디오·이미지 등 다양한 멀티미디어 기록 역시 단일 파일 형태로 생성되었음

- **(데이터 통합 환경에서 원문 특정의 어려움)** 데이터세트(DB형)를 비롯한 다양한 데이터 유형이 통합·공유되는 환경으로 변화하면서, 기록물의 원문이 단일 파일로 존재하지 않고 DB와 응용 내부에 분산된 형태로 저장되는 사례가 증가하고 있음. 이에 따라 통합된 데이터 환경에서는 기록이 최초로 생성될 당시의 형태와 내용을 그대로 담고 있는 '원문'을 특정하는 것이 어렵거나 사실상 불가능해지고 있음. 따라서 기록물을 이관하기 위해서는, 분산된 데이터를 DB로부터 최소 한 번은 추출하여 재구성하는 과정이 전제가 됨

- **(국내 데이터세트 기록관리 구조와 원문 추출의 한계)** 국내 데이터세트(DB형) 기록관리는 DB 전체를 하나의 기록관리 단위로 보지 않고, 업무기능 단위로 DB를 구분하여 기능별로 데이터세트를 부여하는 방식을 채택하고 있음. 이 방식에서는 각 업무기능에 해당하는 '원문' 데이터를 DB로부터 추출해야 하지만, 추출의 난이도는 업무 특성과 데이터 구조에 따라 크게 달라짐. 또한 DBMS마다 제공하는 다른 형식의 dump 파일을 추후 복원을 위해서 제공하고 있음. 이를 원문으로도 사용할 수 있지만, 이를 그대로 사용하는 방식은 국내 기록관리 단위인 데이터세트 구조와 맞지 않을 수도 있음. 따라서 원문을 명확히 특정하기 어려운 경우에는 대체적인 추출 방식이나 보존 전략을 별도로 마련할 필요가 있음

- **(원문 재정의의 필요성과 사례: 인사관리 기능)** 원문 추출이 어려운 경우에는 데이터베이스가 지원하는 업무기능의 본래 목적을 기준으로 기록물 범위를 재정의해야 함. 예를 들어 인사관리 기능의 핵심은 인사정보의 생성·수정 과정이지만, 실제로 관리해야 하는 기록물은 '동결된 인사카드'라는 최종 산출물임. 따라서 이관 대상 기록물은 전체 직원의 인사카드 데이터이며, 화면 표시 또는 증빙으로 활용되는 인사카드 서식이 기록물의 실질적인 원문으로 간주될 수 있음. 즉, DB 내부의 생성·수정용 데이터는 기록물의 본질적 가치와 직접 관련되지 않으며, 최종 결과물인 인사카드와 그 서식이 해당 업무기능의 원문이 됨

○ 국내 데이터세트(DB형) 유형의 전자기록물의 원문의 이해와 제안

- **(절차적·기술적·관리적 요건 충족 시 '대체 원문' 인정 필요)** 미국 NARA와 호주 PROV 사례는 디지털화(Digitization)나 변환(Migration)을 통해 원문을 대체하려면 기술적 기준, 완전성 검증, 절차적·관리적 승인이라는 요건을 반드시 충족해야 한다는 점을 보여줌. 국내 데이터세트(DB형) 기록물에서도 각 DBMS나 DBeaver 같은 일반화된 도구 외에 SQL 기반 추출 방식을 사용할 수 있는데, 이 경우에도 변환 과정의 기술적 기술, 완전성 검증, 절차적·관리적 승인을 충족하면 해당 산출물을 '대체 원문'으로 인정할 수 있어야 함

- **(검증된 도구 기반 dump·SIARD 추출본을 '원문'으로 인정하는 기준 마련)** 데이터세트(DB형) 기록물은 DB 구조·관계·로직 속에 존재하므로, 각 DBMS 표준 기능, DBeaver와 같은 공인 관리 도구, SIARD_KR 도구 등을 이용해 해당 데이터세트를 안정적으로 추출할 수 있다면, 그 결과물(dump 파일 또는 SIARD 파일)을 원문으로 인정하는 것이 필요함. 이러한 파일은 데이터세트 단위와 명확히 대응되고, 필수보존속성을 유지하며, 장기적으로 재현 가능하다는 점에서 원문 대체의 실효성을 확보할 수 있음. 공인 도구 기반 추출 결과물에 대한 설명가능한 방식으로의 변환 기준은 이관·보존 실무의 일관성과 효율성을 높이는 데 기여함

- **(업무기능과 보존속성을 반영한 '본래 목적 기반 원문' 정의 필요)** 데이터세트(DB형) 기록물은 DB와 애플리케이션 내부에 분산된 형태로 존재하므로, '최초 생성 형태의 원문'을 특정하기 어려운 경우가 많음. 특히 국내는 DB 전체가 아니라 업무기능 단위로 데이터세트를 식별하므로, 원문은 기술적 생성 단위가 아니라 해당 기능의 본래 목적을 반영한 최종 기록물로 보아야 함. 예를 들어 인사관리 기능에서는 생성·수정 데이터가 아니라 최종 동결된 인사카드 데이터와 서식이 원문에 해당함

3.3 데이터세트(DB형) 유형의 기록물에서의 보존포맷에 대한 이해

가. 보존포맷 개요

○ 보존포맷의 정의

- 「공공기록물 관리에 관한 법률 시행령」 제2조 제13호에서는 아래와 같이 장기보존패키지와 보존포맷의 개념을 규정함. 장기보존패키지는 전자기록물, 기록관리 메타데이터, 보존포맷, 행정전자서명, 기타 필요한 정보를 하나로 묶은 단위로 정의됨. 보존포맷은 '기록물의 보존을 위한 파일형식'으로 정의하고 있음. 보존포맷은 기록을 오랜 기간동안 안전하게 보호하고 유지하는 '집'의 역할을 한다고 볼 수 있음

> **제 2 조(정의)** 이 영에서 사용하는 용어의 정의는 다음과 같다.
> 13. "장기보존패키지"란 다음 각 목의 자료를 함께 묶은 것을 말한다.
> 가. 전자기록물
> 나. 기록관리 메타데이터
> 다. 보존포맷(기록물 보존을 위한 파일형식을 말한다. 이하 같다)
> 라. 행정전자서명(행정전자서명이 아닌 전자서명을 사용하는 기관의 경우에는 전자서명을 말한다. 이하 같다) 등 진본확인 정보
> 마. 그 밖에 전자기록물 관리에 필요하다고 중앙기록물관리기관의 장이 인정하는 정보

○ 보존포맷 변환(Migration)의 필요성

- 보존포맷으로 변환하는 목적은 기록물이 오랜 시간이 지나도 접근·열람이 가능한 상태를 보장하기 위함임. 생산 당시 파일형식이 이미 보존포맷이면 변환 불필요함. 그러나 대부분의 전자기록물은 다양한 응용프로그램에서 생성되므로, 장기 접근성과 기술 독립성을 확보하기 위해 보존포맷으로의 변환이 필수적임

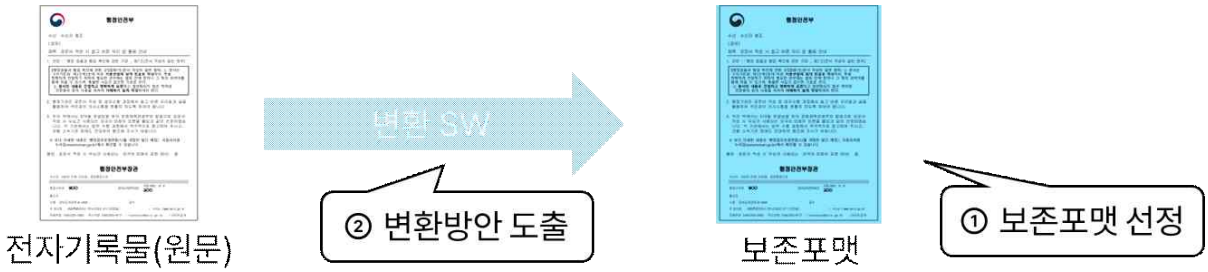
○ 단순 변환의 한계와 필수보존속성

- 단순히 전자기록물을 보존포맷으로 변환하는 것만으로는 충분하지 않음. 보존포맷 변환 과정에서 원문이 가진 필수보존속성(Significant Properties)을 온전히 유지해야 함

- 예를 들어 HWP 공문서를 PDF/A-1로 변환할 경우, 변환 소프트웨어에 따라 보존 품질이 달라짐. 따라서 임의의 변환 SW를 사용하는 것은 부적절하며, 필수보존속성을 완전하게 보존할 수 있는 SW를 선택하거나 새로 개발해야 함. 국가기록원은 이러한 원칙에 따라 문서유형 기록물에 대해서는 문서보존포맷(PDF/A-1) 변환 전용 소프트웨어를 자체적으로 개발해 운영 중임

나. 전자기록물(원문)의 보존포맷 변환(Migration) 과정

○ 전자기록물(원문)을 보존포맷으로 변환하는 과정은 단순한 형식 변경이 아니라, 기록의 진본성과 필수보존속성을 보장하기 위한 절차로 구성됨. 변환과정은 <그림 35>와 같이 ① **보존포맷 선정**과 ② **변환방안 도출**의 두 단계로 구분됨



<그림 35> 전자기록물(원문)의 보존포맷으로의 변환과정

① 보존포맷 선정

- 보존포맷 선정 단계에서는 전자기록물(원문)의 유형을 분석하고, 해당 유형의 필수보존속성(Significant Properties) 을 온전히 유지할 수 있는 보존포맷을 결정함. 보존포맷은 단순히 '열람 가능한 파일형식'이 아니라, 장기보존을 위한 기술적 안정성과 상호호환성을 갖추어야 함

- 이에 따라, 문서, 이미지, 오디오·비디오, 데이터세트 등 유형별로 공공표준(「NAK 37:2022(v1.0)」)을 고려해 보존포맷을 선정함. 이 단계에서의 핵심은 전자기록물(원문)의 구조적·내용적 속성을 보존포맷에서도 동일하게 재현할 수 있는 요소들을 가졌는지를 평가하는 것임

- 예를 들어 문서유형의 전자기록물의 경우, 보존포맷의 유형이 텍스트, 스프레드시트, 프리젠테이션 중 어디에 해당되는지 결정하고, 아래 <표 20>에서 전자기록물(원문)의 필수보존속성을 온전히 담을 수 있는 보존포맷으로 선택함

<표 20> 미국 NARA의 매체 변환 원문 폐기 관련 사항

구분	텍스트형	프리젠테이션형	스프레드시트형
보존포맷	PDF,DOCX, PDF/A-1, PDF/A-2, ODT, EPUB	PDF, PDF/A-1, PDF/A-2, PPTX, ODP	XLSX, ODS
수용가능포맷	HWPX	-	-

출처: 「NAK 37:2022(v1.0)」

② 변환방안 도출

- 보존포맷이 결정된 이후에는 전자기록물의 필수보존속성을 최대한 온전히 유지할 수 있는 변환 절차와 기술적 방안을 마련함. 변환방안 도출의 핵심은 전자기록물(원문)의 필수보존속성(내용, 구조, 맥락, 기능, 외관)이 보존포맷이 가진 요소를 통해 완전하게 재현되도록 변환하는 것임. 변환은 단순한 파일 형식 변경이 아니라, 원문이 지닌 의미·구조·표현적 특성을 손상 없이 이전하는 것을 목표로 함

- 따라서 변환과정에서는 소프트웨어별 변환 품질을 검증하고, 필요시 새로운 변환 알고리즘이나 전용 도구를 개발해야 함. 예를 들어, HWP 형식의 공문서를 PDF/A-1로 변환할 때는 서체, 여백, 표 구조, 하이퍼링크, 메타데이터 등의 속성이 손실되지 않도록 해야 함. 국가기록원에서는 이러한 품질 기준을 충족하기 위해 자체 개발한 PDF/A-1 변환 도구를 활용하고 있는 것이 장기보존 품질 확보의 좋은 사례라고 볼 수 있음

다. 데이터세트 유형 전자기록물(원문)의 보존포맷 변환 과정

○ DB형 데이터세트 전자기록물의 보존포맷 변환에서의 어려움

- DB 데이터는 실시간 검색과 열람을 위해 실행 중인 DBMS 내에서만 존재함. 따라서 이를 원문의 파일포맷에서 다른 파일포맷으로 변환하는 대상으로 보기 어려움. 또한 DB 데이터와 그 구성요소들은 서로 복잡하게 연계되어 있어 단일 파일포맷으로 변환하기도 쉽지 않음

- 유럽 표준으로 제정된 SIARD는 2023년도 국가기록원 R&D 연구과제 「시청각 및 데이터세트 유형 보존포맷 선정기준 개발 및 전자기록물 보존포맷 선정체계 구축 연구」에 따르면 보존포맷이 아닌 수용가능포맷에 해당함 (<표 21> 참조)

<표 21> 데이터세트-데이터베이스형 권고포맷

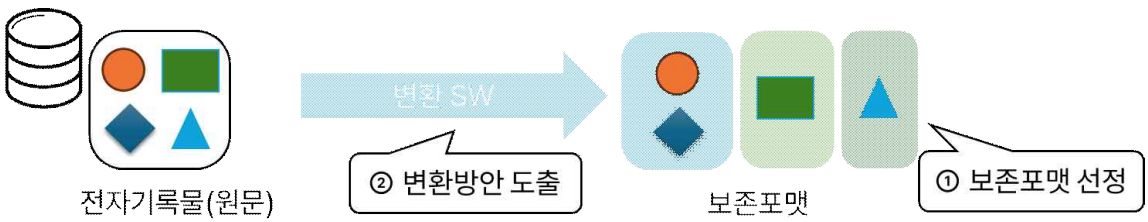
보존포맷	수용가능포맷
CSV, XLSX, XML, JSON	SIARD

출처: 「NAK 37:2022(v1.0)」

- 국내의 데이터세트형 기록관리 단위(데이터세트)를 기준으로 운영하기에는 구조적 차이가 존재하며 지원가능한 DBMS 종류도 제한적임. 이로 인해 DB형 데이터세트 전자기록물의 보존포맷 변환에는 기술적·구조적 제약이 따름. 따라서 DB형 데이터세트 전자기록물의 보존포맷 변환은 다음과 같은 방식으로 진행되는 것이 타당함

① 보존포맷 선정

- DB형 데이터세트의 경우에서도, 보존포맷 선정은 전자기록물(원문)의 필수보존속성을 온전히 담을 수 있는 형식을 선택하는 것을 원칙으로 함. 다만, 하나의 보존포맷으로 DB형 데이터세트를 완전하게 변환하기 어려운 경우에는, <그림 36>과 같이 여러 보존포맷을 활용해 다수의 파일로 구성할 수 있음



<그림 36> DB형 데이터세트 유형 전자기록물의 보존포맷으로의 변환과정

② 변환방안 도출

- 변환방안 도출 단계에서는 필수보존속성을 완전하게 변환할 수 있는 단일 변환 소프트웨어를 선정하기 어려움. 따라서 선정된 다수의 보존포맷들을 조합하여 DB 데이터와 관련 필수보존속성들을 온전하게 변환할 수 있는 기술적 방안과 구성 방법을 마련하는 것이 필요함

- 이 과정에서는 데이터 구조, 관계정보, 스키마 정의 등 DB 고유의 속성이 손실되지 않도록 변환 절차를 설계하는 것이 중요함. 또한 변환방안은 공식 문서로 정리하여 명확히 공표해야 함

3.4 데이터세트의 이관·보존을 위한 새로운 보존포맷 형태 도출 및 규격 작성

가. 데이터세트(DB형) 유형 보존포맷의 설계 원칙

○ DB형 데이터세트 유형 기록물을 위한 오픈포맷 기반 보존포맷 설계 원칙

- DB형 데이터세트의 장기보존을 위한 오픈포맷 기반 보존포맷 설계는, 기존의 한계를 보완하고 기록으로서의 본질적 속성을 유지하기 위한 체계적 기준을 마련하는 데 목적이 있음. 이 설계 원칙은 ① 기록의 4대 속성 보장, ② 기존 SIARD의 한계 해결, ③ 데이터로서의 속성 활용이라는 세 가지 관점에서 설정됨

<표 22> DB형 데이터세트를 위한 오픈포맷 기반 보존포맷 설계 원칙

구분	원칙
기록의 4대 속성 보장	① DB 데이터의 구조와 내용을 온전히 표현할 수 있어야 한다. ② 시스템 환경에 상관없이 데이터의 접근성, 가독성, 호환성을 높여야 한다. ③ 재현을 위해 DB 데이터와 밀접하게 연관된 다른 정보를 함께 저장해야 한다. ④ 무결성을 위한 도구 및 정보를 내재화한다.
기존 SIARD 문제 해결	⑤ 국내 데이터세트 관리 기준인 데이터세트 단위로 기록관리할 수 있어야 한다. ⑥ 국내외 대부분 DBMS에도 적용할 수 있어야 한다. ⑦ 보존포맷은 전용 소프트웨어 없이도 생성할 수 있어야 한다.
데이터로서의 활용	⑧ 데이터 처리와 분석이 용이하도록 구조화된 데이터 형식으로 저장한다.

- 기록의 4대 속성 보장

① 새로운 보존포맷은 데이터베이스의 구조와 내용을 온전히 표현할 수 있어야 함. 이를 위해 데이터는 국제 오픈표준이면서 보존포맷인 XML 형식으로 저장하여 구조적 일관성과 표현의 명확성을 확보함

② 데이터는 시스템 및 네트워크 환경에 상관없이 이용자가 쉽게 접근, 열람할 수 있어야 함. 이를 위해 UTF-8 인코딩 방식을 적용하였고, 보존포맷 내부 및 외부의 파일들을 위치를 명시하기 위해서 URI 방식을 사용함. 단, 내부 및 외부 파일들이 대량, 대용량일 경우 archive 형식(보존포맷에서는 zip만 허용)으로 묶였을 경우에도 zip 내부의 개별 파일을 지정할 수 있도록 zip 내부 경로 표기 방식을 허용함

상황	보존포맷의 attachment/ 폴더 아래에 **attach_bundle.zip**이 있고, 그 안에 cert/cert_A.hwp 파일이 있을 경우
zip 내부 경로 표기 방식 예시	attachment/**attach_bundle.zip!**/cert/cert_A.hwp

③ 본 연구에서 제안하는 보존포맷은 데이터베이스(DB)의 데이터를 전제로 함. 기록 관점에서는 데이터만 중요할 수도 있지만, 경우에 따라 데이터 구조나 DBMS의 기능까지 함께 보존해야 하는 상황도 존재함. 따라서 향후 DBMS 환경에서 기록을 재현할 수 있는 상황까지 고려해야 하며, 이를 위해 데이터와 밀접하게 연관된 Routine(Trigger, Stored Procedure, Function 등), Event, View, Index, User 등의 구성 요소도 함께 보존할 수 있어야 함

④ 무결성을 검증할 수 있도록 필수적으로 암호화 해시값을 내재화하여 데이터 변조 여부를 확인함. 또한, 보존포맷이 기록의 구조가 일관되고 표준화된 방식으로 생성되었음을 확인하게

위해 XSD를 내부 구조를 검증함

- 기존 SIARD의 한계 해결

⑤ 기존 SIARD는 데이터베이스 전체를 하나의 단위로 묶어 관리하는 구조이기 때문에, 테이블보다 더 세분화된 단위로 기록을 관리하기 어려움. 새롭게 제안된 SIARD_KR은 테이블 수준까지 관리할 수 있도록 개선되었으나, 여전히 국내 기록관리 기준에서 요구하는 '데이터세트 단위'의 기록관리를 하기에는 한계가 있음. 따라서 데이터는 행과 열 수준을 넘어, 더 유연하게 관리할 수 있는 방안이 필요함

⑥ 또한 SIARD는 제한된 6종의 DBMS만 지원하여 국내에서 널리 사용되는 국산 DBMS 적용에 제약이 있었음. 새로운 오픈포맷 기반 보존포맷은 국내외 대부분의 DBMS 환경에서도 적용할 수 있도록 설계함

⑦ 기존 SIARD는 전용 소프트웨어(SIARD Suite)에 대한 의존도가 높아 공공기관의 활용성과 접근성이 낮음. 이에 따라 새로운 보존포맷은 전용 SW 없이 생성할 수 있도록 설계함

- 데이터로서의 활용

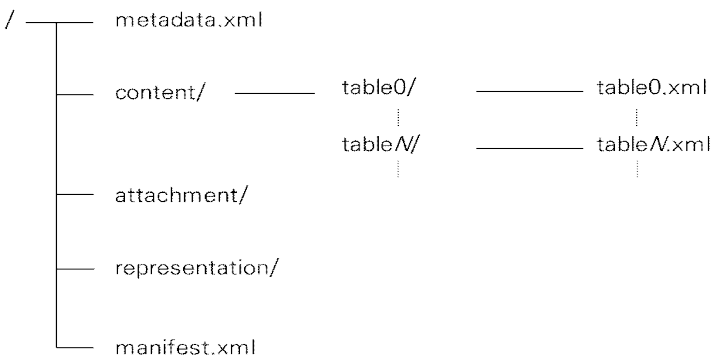
⑧ DB형 데이터세트는 단순히 읽히는 기록이 아니라, 컴퓨터에 의해 효율적으로 처리될 수 있는 데이터로서의 속성을 가져야 함. 이를 위해 모든 데이터는 구조화된 데이터 형식으로 저장하여, 자동 처리 및 데이터 분석에 활용할 수 있는 기반을 마련함

나. 데이터세트(DB형) 유형의 오픈포맷 기반 보존포맷 구조 설계

○ 본 절에서는 보존포맷의 구조와 역할을 개괄적으로 기술함. 보존포맷에 대한 구체적인 사양은 '[첨부04] 데이터세트(DB형) 유형 보존포맷 규격 표준(안)'을 참고

○ 보존포맷의 구성요소와 디렉토리 구조

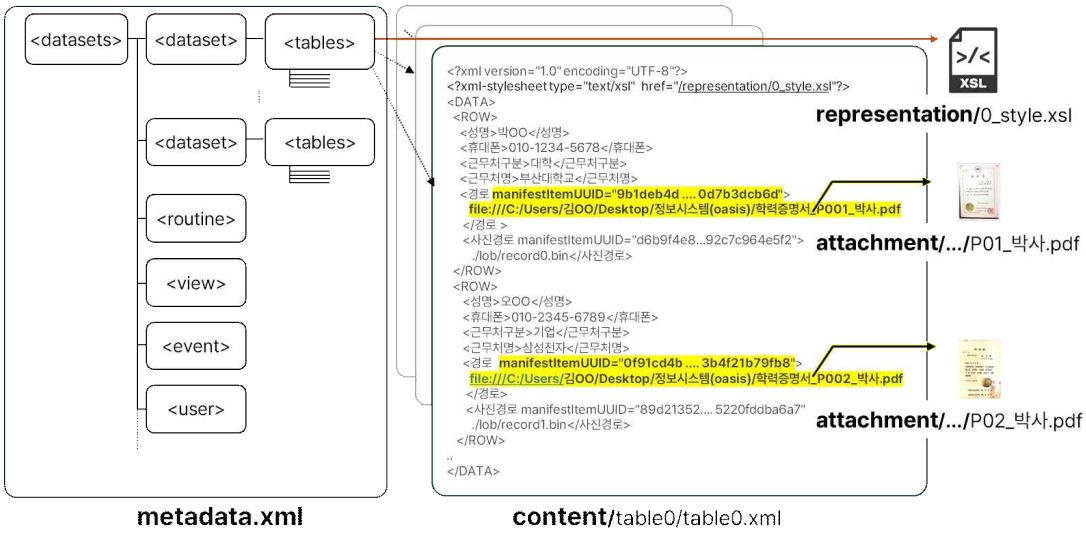
- DB형 데이터세트의 보존포맷은 여러 구성요소와 폴더로 이루어짐. 보존포맷은 **① 스키마와 기타요소**, **② 실제 데이터**, **③ 붙임 파일**, **④ 재현 정보**, **⑤ 구성목록**으로 구성되며, 각 요소는 <그림 37>과 같이 폴더 구조 내에 지정된 위치에 배치됨



<그림 37> 데이터세트 유형 보존포맷의 내부 구조

① 스키마와 기타요소: metadata.xml

· metadata.xml은 데이터세트(DB형) 보존포맷의 핵심 메타데이터 파일로, 보존포맷의 기본 데이터 단위인 테이블(Table) 정보뿐 아니라 트리거(Trigger), 저장 프로시저(Stored Procedure), 뷰(View), 인덱스(Index) 등 데이터세트 전체를 구성하는 스키마 및 기타 요소에 관한 정보를 포함함



<그림 38> metadata.xml 와 다른 구성요소들과의 관계

· metadata.xml의 구조는 <그림 38>과 같이 root 요소로 <datasets>를 갖음. <datasets> 요소에서 언급되는 내용들은 하나의 정보시스템 내 단일 DBSM에서 관리되고 있는 데이터를 전제로 함. 실제 데이터의 '스키마'와 관련된 하나 이상의 <dataset> 요소와 '기타요소'와 관련된 <routine>, <view>, <event>, <user> 요소로 구성됨. 또한 <datasets> 요소에는 보존포맷 생성 시점을 나타내는 <archivalDate>와 DBMS 이름과 버전을 기록하는 <database> 요소도 포함됨. <그림 38>은 metadata.xml이 보존포맷의 다른 주요 요소들과 어떻게 연계되는지를 보여줌. metadata.xml은 <tables> 및 하위 <table> 요소에서 각 테이블이 어떤 방식으로 생성되었는지를 DDL 명령문으로 기술하며, 실제 데이터를 저장하는 content/ 폴더와 해당 테이블의 재현 정보를 담는 representation/ 폴더를 연결하는 중심 역할을 함

· 본 연구에서 제시하는 오픈포맷 기반 보존포맷은 '단위 업무'를 중심으로 식별하는 국내 행정정보 데이터세트 기록관리 기준에 적용할 수 있으며, 생산 형태(DB) 그대로의 보존도 가능하도록 설계함

· metadata.xml의 <dataset> 요소는 국내 기록관리 기준에서 정의하는 '데이터세트' 단위에 대응함. 데이터세트는 DB 전체가 아니라 특정 '단위 업무'와 관련된 데이터 묶음을 의미하며, 여러 개의 테이블과 선택된 컬럼들로 구성될 수 있음. 또한 생산 형태(DB) 그대로 보존하는 경우, <dataset>은 여러 테이블을 포함하는 DB 관리 단위를 의미함. 예를 들어 Oracle, MySQL, MS SQL Server, 티베로에서는 Schema, 그리고 큐브리드에서는 Database가 이러한 관리 단위에 해당하며, 이는 모두 <dataset> 요소와 대응됨. <dataset>은 하나의 <tables> 요소와 기타 부가 요소로 구성될 수 있음. 기타 요소는 현재 표준으로 정하지 않았고, 사용자 정의(User-defined) 방식에 따라 자유롭게 확장하여 추가할 수 있음. <tables> 요소는 하나 이상의 <table> 요소로 이루어짐.

· <table> 요소는 하나의 <dataset>를 구성하는 각 테이블과 특정 컬럼들이 어떤 방식으로 생성되었는지를 설명하는 요소임. 각 <table>의 데이터는 '① 생산형태(DB) 그대로 보존'하는 데이터로 생성될 수 있고, '② 서식형태 보존을 위해 가공'된 데이터로 생성될 수도 있음. 이 밖에도 다양한 방식으로 만들어질 수 있음. 본 과제에서는 '① 생산형태(DB) 그대로 보존'과 '② 서식형태 보존을 위해 가공'된 두 가지 경우의 <table> 구조를 정의하였고, 그 밖의 방식은 사용자 정의 방식으로 자유롭게 확장·추가할 수 있도록 설계함

· <routine> 요소는 stored procedure, function, trigger 등 특정 조건과 입력에 따라 동작을 수행하는 요소를 보관하는 용도로 사용되며, 각 루틴을 생성하는 DDL(Data Definition

Language) 명령문을 CDATA 형태로 기록함

· <view> 요소는 DBMS에서 사용되는 View 객체를 저장하기 위한 XML 요소임. <tables> 생성 과정에서 View를 함께 확인할 수 있으므로, View 생성과 관련된 DLL 명령어를 CDATA 형태로 이 요소에 기록함

· <event> 요소는 DBMS의 Event 객체를 생성하는 DDL 명령어를 CDATA 형태로 기록하기 위한 요소임

· <user> 요소는 데이터베이스 사용자(User) 계정 및 권한과 관련된 DDL 명령문을 CDATA 형태로 저장하기 위한 요소임

· <routine>, <view>, <event>, <user> 등과 같은 기타 요소들은 모두 CDATA 형식으로 DDL 명령문을 기록하면 되는 항목이므로, 반드시 각각을 별도로 구분하여 기록할 필요는 없음. 따라서 필요에 따라 개별 요소로 기술할 수도 있고, <dbobj> 요소 안에 하나의 범주로 묶어 여러 요소를 함께 포함해 기록할 수도 있음

② 데이터: content/table*N*/table*N*.xml (*N* = 0, 1, ...)

· 실제 데이터를 저장하는 공간으로, 각 테이블은 metadata.xml의 <table> 요소의 datapath 속성과 연결되어 있음. 테이블마다 table*N*/ 폴더를 생성하고, 그 안에 table*N*.xml 파일을 저장함. 여기서 *N*은 테이블 번호를 의미하며, 식별을 위해 0부터 순차적으로 증가하는 방식으로 번호를 부여(예: DB에서 테이블이 생성한 순서대로 번호 지정)

· '**일반 데이터**'는 UTF-8 형식의 '행 기반 XML 구조'로 table***N***.xml 파일에 저장함. '행 기반 XML 구조'는 DB의 한 행(row)을 <ROW> 요소 하나로 표현하고, 테이블의 행들을 순차적으로 XML로 기록하는 방식임. <ROW> 내부에는 해당 행이 가진 각 컬럼을 XML 하위 요소로 변환하여 배치하며, 하위 요소의 태그 이름은 컬럼명, 태그 내용은 컬럼값이 1:1로 대응됨. 이를 통해 관계형 데이터의 구조가 XML 내에서 직관적이고 일관된 형태로 재현될 수 있음

· '**LOB(Large OBject) 유형의 대용량 데이터(BLOB(Binary LOB), CLOB(Character LOB))**'는 아래의 1) Base64 인코딩 방식과 2) lob 폴더 저장 방식 중 하나로 보존포맷에 포함할 수 있으며, 1)번 방식이 기본(default) 방식임

  1) 이 방식은 기본 저장 방식으로, 별도의 지시나 특별한 문제가 없는 경우 그대로 적용함. 데이터를 저장할 때 Base64로 인코딩하여 table*N*.xml 내부에 포함함. 이러한 기능은 대부분의 DBMS 전용 애플리케이션에서 기본적으로 제공함

  2) LOB 데이터는 content/table*N*/lob/ 폴더에 별도로 저장되며, BLOB은 recordM.bin, CLOB은 recordM.txt 형식으로 파일명을 부여함. 여기서 M은 해당 테이블에서 LOB 객체가 생성된 순서를 의미하며 0부터 순차적으로 번호를 매김(예: table1의 세 번째 BLOB은 record3.bin). Base64 데이터가 있던 필드에는 lob/ 폴더의 상대경로와 manifestItemUUID가 table*N*.xml에 기록되며, manifest.xml에는 manifestItemUUID, 해시 알고리즘, 생성 시간, 해시값, 경로명이 함께 저장됨

· **'붙임 파일'** 데이터도 일반데이터와 동일하게 DB에 저장된 값(해당 값은 붙임 파일의 원래 저장되었던 외부저장소의 경로명)을 그대로 table*N*.xml에 기재함. 이때, 보존포맷 내에서 해당 파일을 식별하기 위해 요소의 속성인 manifestItemUUID에 UUID를 부여함. manifestUUID는 보존포맷의 모든 구성요소를 관리하는 manifest.xml에서 사용하는 식별자임

③ 붙임 파일: attachment/

· HTML이나 웹프로그래밍 환경에서는 파일을 직접 열거나 하이퍼링크로 연결해 접근할 수 있음. 그러나 LOB 파일은 먼저 파일 형태로 추출해야만 접근이 가능하므로, 파일을 열기까지 여러 단계를 거쳐야 하는 번거로움이 있음. 이 때문에 국내 많은 DB의 파일들은 LOB 형태로 저장하지 않고, DB 필드에는 파일의 위치(경로명, URL 등)만 기록하고 실제 파일은 해당 위치가 가리키는 DB 외부의 저장소에 보관함. 이러한 파일을 '붙임 파일'이라 하며, 이를 보존

포맷 내에 포함하기 위해 attachment/ 폴더에 저장함

· 다만, 대량·대용량 등의 기술적 사유로 인해 attachment/ 폴더가 아니라 외부 저장소에 보관해야 하는 경우도 있음. 각 파일은 content/table*N*/table*N*.xml 및 manifest.xml과 연계되어야 함. 이를 위해 파일마다 해당 파일의 정보를 담고 있는 요소에 manifestItemUUID 속성을 생성하고 여기에 UUID를 부여함

④ 재현 파일: representation/

· 데이터베이스의 시각적·논리적 구조를 복원하거나 이용자에게 제공하기 위한 재현 정보와 관련된 파일이 포함될 수 있으며, 이러한 파일은 representation/ 폴더에 저장됨. 재현 정보는 해당 데이터와 연계되어야 하므로, metadata.xml의 각 <table> 요소 아래에 <representationInfo> 요소를 두고, 재현 정보 파일의 경로명을 기록함

· <representationInfo>는 하나의 <description> 요소와 하나 이상의 <location> 요소로 구성됨. method 속성을 통해 재현 정보의 유형(xml_xslt, installer, executable 등)을 지정할 수 있음

⑤ 구성목록: manifest.xml

· manifest.xml은 보존포맷을 구성하는 모든 파일에 관련된 항목들을 목록화하여 관리할 수 있도록 하는 역할을 함. <manifest>를 root 요소로 갖고 있으며, 하나 이상의 <item> 요소를 하위 요소가 포함하고 있음. 각 <item>은 보존포맷을 구성하는 하나의 파일에 대응됨

· <item> 요소는 hashAlgorithm, createDateTime, manifestItemUUID 속성을 가짐. hashAlgorithm은 암호화 해시 방식(SHA2-256, SHA2-512, SHA-1, MD5 등)을 저장하고, createDateTime은 파일을 생성한 시간을 ISO 8601 날짜·시간 표기 형식으로 기록함.

· manifestItemUUID는 보존포맷을 구성하는 파일들을 구분하는 식별자 역할을 하며, 패턴화되고 정규화된 식별체계 없이 항목들을 구분할 수 있 UUID 방식으로 식별자를 부여함

· <item> 요소는 하나의 <hashValue>와 하나의 <location> 요소로 구성됨. <hashValue>는 암호화 해시값을 저장하고, <location>은 각 파일의 보존포맷 내부 또는 외부의 위치를 URI 형식으로 저장함. 단, 내부 및 외부 파일들이 대량, 대용량일 경우 archive 형식(보존포맷에서는 zip만 허용)으로 묶였을 경우에도 zip 내부의 개별 파일을 지정할 수 있도록 zip 내부 경로 표기 방식을 허용함

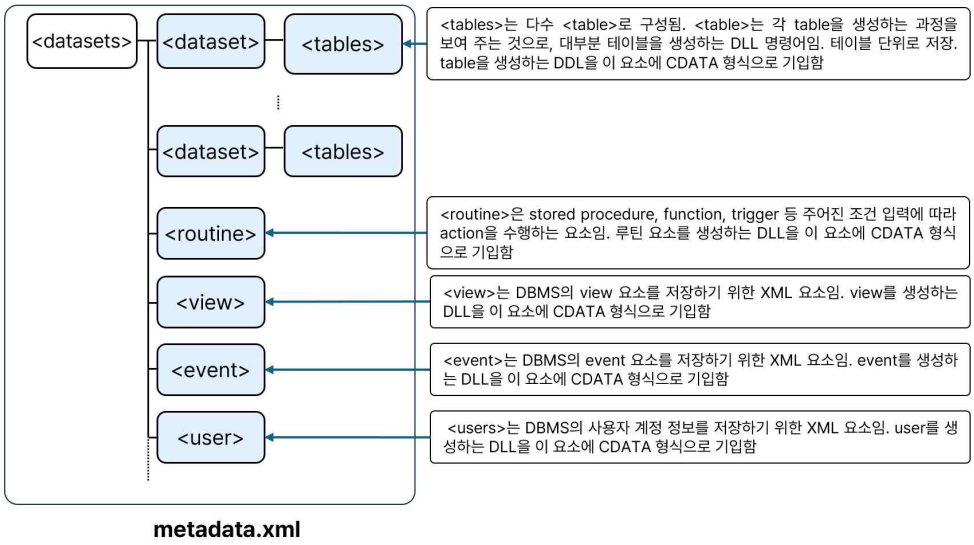
· 본 연구에서 설계한 보존포맷은 붙임파일이 보존포맷 외부에 저장되는 경우도 허용함. 따라서, 파일의 위치정보가 보존포맷 내 여러 파일에 중복 저장되면 관리가 복잡해지므로 실제 파일의 위치 정보는 manifest.xml에서만 관리하고, 다른 파일에서는 manifestItemUUID를 통해 해당 파일과 연계하는 방식으로 설계함

다. 데이터세트(DB형) 유형의 보존포맷 생성 절차

○ 첫 번째 절차: metadata.xml 생성

[metadata.xml 생성 개요]

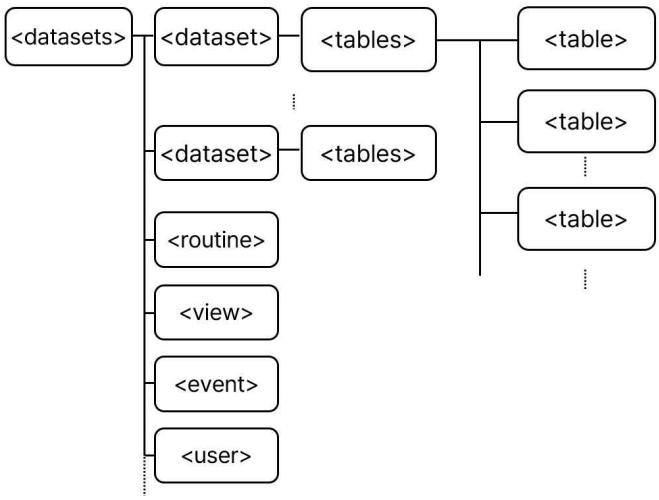
- metadata.xml의 구조는 <그림 39>와 같이 요약됨. <datasets>는 하나 이상의 <dataset>으로 구성되며, 각 <dataset>은 <tables>, <routine>, <view>, <event>, <user> 요소로 이루어짐. 이 중 <tables>는 하나 이상의 <table>을 포함하며, 실제 데이터 파일의 위치를 지정하고 해당 테이블의 데이터가 생성된 방식을 설명함. 생성 방식에 따라 '① 생산형태(DB) 그대로 보존'으로 생성될 수 있고, '② 서식형태 보존을 위해 가공'된 데이터로 생성될 수도 있으며, 이밖에도 다양한 방식으로 만들어질 수 있음. <routine>, <view>, <event>, <user>는 실제 데이터 이외의 구성요소로, DBMS에서 동일하게 생성할 수 있도록 해당 객체의 DDL 명령어를 저장함



<그림 39> metadata.xml 생성 요약

<datasets>

- metadata.xml은 <그림 40>과 같은 구조의 요소(element)들로 구성되어 있음. <datasets>는 최상위 요소이고, 다수의 <dataset>와 <routine>, <view>, <event>, <user>를 포함할 수 있음. 하나의 <dataset> 요소는 국내 데이터세트 유형의 기록관리 단위인 '데이터세트'에 해당함



<그림 40> metadata.xml의 element 구조

<dataset>

- <dataset>는 <tables>구성요소로 포함하고 있음

<tables>

- <tables>는 하나 이상의 <table>로 구성되며, 각 <table>은 하나의 테이블 정보와 연결됨. 일반적으로 DBMS로부터 테이블 생성 관련 DDL를 추출할 경우, 실제 테이블을 생성하는 DDL을 실행하기 전과 실행한 후에 환경을 설정하는 명령어를 실행함. 이를 저장하기 위해서 <tables>에 <sesseionDDLstart>와 <sessionDDLend> 요소를 선택적으로 포함할 수 있음

<table>

- 하나의 <table> 요소는 반드시 하나의 content/table*N*/table*N*.xml (*N* = 0, 1, ...) 파일과 연결됨. 각각의 <table>은 대응되는 table*N*.xml이 어떻게 만들어졌는지 설명하는 요소임

- table*N*.xml은 '① 생산형태(DB) 그대로 보존'하는 테이블을 의미할 수도 있으며, '② 서식형태 보존을 위해 가공'된 테이블일 수도 있음. table*N*.xml의 생성 방식에 따라 <table>요소의 하위 요소와 속성이 달리 설정됨

① 생산형태(DB) 그대로 보존 → DB 내의 개별 테이블

· <table> 요소에서 datapath 속성(필수)은 반드시 존재해야 하며, 해당 테이블의 실제 데이터의 위치를 URI 형태로 명시함. name 속성(필수)은 이 경우 해당 테이블 명은 존재하므로 명시해야 하며, sourceType(필수)은 DB 내의 개별 테이블이므로 'original'로 설정함

· <table>의 요소값은 해당 테이블을 생성하는 DDL을 <sources>라는 요소에 입력함. DDL을 저장할 때는 특수문자나 예약어가 포함될 수 있으므로 CDATA 방식으로 저장함

· 생산형태(DB) 그대로 보존일 경우, <table> 요소의 예시는 아래와 같이 제시할 수 있음

```
<table datapath="/content/table0/table0.xml" name="개인기본" sourceType="original">
    <sources> <![CDATA[
    -- Table structure for table `개인기본`
    DROP TABLE IF EXISTS `개인기본`;
    /*!40101 SET @saved_cs_client = @@character_set_client */;
    /*!50503 SET character_set_client= utf8mb4 */;
    CREATE TABLE `개인기본` (
    `대표개인번호` varchar(36) COLLATE utf8mb4_unicode_ci NOT NULL,
    ...
    `수정일시` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE
    CURRENT_TIMESTAMP,
    PRIMARY KEY (`대표개인번호`),
    UNIQUE KEY `uk_person_rrn` (`주민등록번호`),
    ) ENGINE=InnoDBDEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_unicode_ci;
    /*!40101 SET character_set_client= @saved_cs_client */;
      ]]>
    </sources>
</table>
```

② 서식형태 보존을 위해 가공 → SELECT 문을 통해 데이터를 추출해 생성된 테이블

· <table> 요소에서 datapath 속성(필수)은 반드시 존재해야 하며, 해당 테이블의 실제 데이터의 위치를 URI 형태로 명시함. name 속성(필수)도 다른 <table>의 name과 구별되도록 명시해야 함. sourceType(필수)은 SELECT 문으로 생성되었으므로 'derived'로 설정함

· <table>의 하위 요소인 <derivation> 요소(필수)에는 derived 방식을 기입하고, <sources> 요소(필수)에는 derived 방식의 대상 데이터를 명시함. 예를 들어, <derivation>의 method 속성(필수)에 "SQL"로 명시하면, <derivation> 요소값에는 해당 SQL 문을 CDATA 방식으로 기입함. 그리고, <sources>에는 SQL과 연관된 테이블들의 DDL을 CDATA 방식으로 기입함

· <derivation>의 method 속성은 derivation 방식은 명시하는 것으로 SQL 이외에 다른 방식도 명시할 수 있음. 예를 들어, "Python", "R", "ETL[1]", "Manual" 등을 들 수 있음

· <sources>에는 derivation의 대상이 되는 원데이터 등을 명시하는 요소임

· <representation> 요소(해당시 필수)는 <table> 데이터와 연계된 재현 도구(xsl/xml 등)가 존재하는 경우 그 관련 정보를 명시하는 요소임. <description> 요소(필수)는 재현 도구에 대한 개요를 기록하며, <location> 요소(해당시 필수)는 재현 도구의 경로명을 나타냄. 이때 재현 도구가 representation/ 폴더 내부에 있을 수도 있고, 외부 저장소에 존재할 수도 있으므로, 경로는 URI 형식으로 작성해야 함

1) Extract-Transform-Load 도구 (예: Pentaho, Talend, Informatica) 기반 변환

· 서식형태 보존을 위해 가공된 경우, <table> 요소의 예시는 아래와 같이 제시할 수 있음

```
<table datapath="./content/table16/table16.xml" name="경력증명서" sourceType="derived">
      <derivationmethod="SQL"> <![CDATA[
            SELECT CONCAT(  …  FROM 개인기본 p WHERE p.대표개인번호 = 'P006';
      ]]> </derivation>
      <sources> <![CDATA[
         CREATE TABLE `경력` (
          `경력ID` bigintNOT NULL AUTO_INCREMENT,
            …
         CONSTRAINT `경력_ibfk_2` FOREIGN KEY (`파일ID`) REFERENCES `첨부파일` (`파일ID`))
      ]]> </sources>
      <representationInfo method="xml_xslt">
          <description> XML과 XSLT를 이용해 table16 데이터를 HTML로 재현하는 방식  </description>
          <location> ./representation/0_style.xslt </location>
          <location> ./representation/1_style.css </location>
      </representationInfo>")
</table>
```

<routine>

- <routine> 요소(해당 시 필수)는 루틴(routine)이 존재하는 경우에만 명시됨. 다만 루틴이 있더라도 <dbobj> 요소에 포함하여 기록하는 경우에는 별도의 <routine> 요소를 생성하지 않음. 루틴은 데이터베이스 내에서 미리 정의해 두고 반복적으로 실행할 수 있는 명령 집합을 말함. 즉, 자주 사용하는 SQL 명령이나 복잡한 처리 로직을 하나의 이름으로 정의해 두고 필요할 때마다 호출할 수 있도록 하는 기능임. 이러한 routine의 종류로는 저장 프로시저(Stored Procedure), 함수(Function), 트리거(Trigger)가 있음. <routine>요소는 이러한 루틴에 해당되는 DDL(예. 'CREATE PROCEDURE', 'CREATE FUNCTION', 'CREATE TRIGGER'로 시작하는 명령어 등)을 저장함

- <routine> 요소의 예시는 아래와 같이 제시할 수 있음

```
<routine> <![CDATA[
   CREATE TRIGGER `trg_첨부파일_bi`
   BEFORE INSERT ON `첨부파일`
   FOR EACH ROW
   BEGIN
     SET NEW.첨부파일경로 = CONCAT(
       'C:₩₩Users₩₩박경환₩₩Desktop₩₩대상정보시스템(oasis)₩₩',
       NEW.첨부파일명   );
   END;
          …
]]> </routine>
```

<view>

- <view> 요소(해당 시 필수)는 뷰(view)가 존재하는 경우에만 명시됨. 다만 뷰가 있더라도 <dbobj> 요소에 포함하여 기록하는 경우에는 별도의 <view> 요소를 생성하지 않음. 뷰는 실제 테이블들을 결합하여 가상적으로 생성된 테이블 형태의 데이터임. 데이터 자체는 이미 원본 테이블에 저장되어 있으므로, 뷰 요소에는 뷰를 생성하는 DDL(예. 'CREATE VIEW'로 시작하는 명령어 등)만 보존하면 됨. 이때 원본 테이블은 반드시 저장되어야 함

- <view> 요소의 예시는 아래와 같이 제시할 수 있음

```
<view> <![CDATA[
   CREATE VIEW `vw_재직자현황` AS SELECT
    1 AS `인사기본ID`,
    1 AS `대표개인번호`,
    1 AS `근무부서`,
    1 AS `직급`,
    1 AS `재직상태구분`,
    1 AS `성명`,
    1 AS `이메일`
]]> </view>
```

<event>

- <event> 요소(해당 시 필수)는 이벤트(event)가 존재하는 경우에만 명시됨. 다만 뷰가 있더라도 <dbobj> 요소에 포함하여 기록하는 경우에는 별도의 <event> 요소를 생성하지 않음. 이벤트는 특정 시간이나 주기에 따라 자동으로 실행되는 SQL 작업으로, 데이터베이스의 예약 기능(스케줄러 역할)을 수행하는 요소임. 이 요소는 event 관련 DDL(예. 'CREATE EVENT'로 시작하는 명령어 등)을 저장함

- <event> 요소의 예시는 아래와 같이 제시할 수 있음

```
<event> <![CDATA[
    CREATE EVENT `ev_호봉_월간보정` ON SCHEDULE EVERY 1 MONTH
      STARTS '2025-10-22 00:00:00' ON COMPLETION NOT PRESERVE
    ENABLE
    DO      BEGIN      CALL oasis.sp_호봉_다음승급일_계산();      END;
    ]]>
]]> </event>
```

<user>

- <user> 요소(해당 시 필수)는 사용자(user)가 존재하는 경우에만 명시됨. 다만 사용자가 있더라도 <dbobj> 요소에 포함하여 기록하는 경우에는 별도의 <user> 요소를 생성하지 않음. 사용자 및 권한 테이블들의 정보들을 포함하고 있는 DDL/DCL(Definition Control Language)(예. 'CREATE USER' 또는 'INSERT INTO user' 시작하는 명령어, 'GRANT'로 시작하는 명령어 등)을 저장함

```
<user> <![CDATA[
    CREATE TABLE `user` (
     `Host` char(255) CHARACTER SET ascii COLLATE ascii_general_ci NOT NULL DEFAULT '',
     `User` char(32) COLLATE utf8mb3_bin NOT NULL DEFAULT '',
     utf8mb3_general_ci NOT NULL DEFAULT 'N',
      ...
     `User_attributes` json DEFAULT NULL,
     PRIMARY KEY (`Host`,`User`)
     INSERT INTO `user` VALUES
     ('%','testuser','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N'
     ,'N','','','','',0,0,0,0,'mysql_native_password','*6BB4837EB74329105EE4568DDA7DC67ED2CA2AD9','N','2
     023-11-01 13:14:32',NULL,'N','N','N',NULL,NULL,NULL,NULL),
      ...
    )
    ]]>
]]> </user>
```

○ 두 번째 절차: content/ 생성

- **[개요]** content/ 폴더에는 실제 DB 데이터가 저장된 파일들이 위치하며, 그 구조는 <그림 42>와 같음. content/ 폴더 하위에는 table*N*(*N* = 0, 1, …) 형태로 폴더가 순차적으로 생성되며, 각 폴더명은 인덱스가 1씩 증가함. 또한 각 table*N* 폴더 내부의 table*N*.xml 파일에는 해당 테이블의 실제 데이터를 저장하고 있음. table*N*.xml 은 문자, 숫자, 날짜 등의 **① 일반 데이터**, **② LOB 데이터**, **③ 붙임 파일** 3가지 유형의 데이터를 기록할 수 있음

  ① 일반 데이터: 문자, 숫자, 날짜 등의 데이터로 '행 기반 XML 구조'로 저장함. 최상위 요소는 <DATA>이며, 다수의 <ROW>로 구성됨. <ROW>의 하위 요소명은 모두 컬럼명이고, 요소 값은 DB에 저장되어 있던 필드값으로 설정됨. 이러한 '행 기반 XML 구조' 방식은 MySQL, MS SQL Server, Oracle 등 대부분 DBMS에서 지원하고 있음. <그림 41>은 행 기반 XML 구조 방식으로 작성된 table0.xml의 사례를 보여줌

<?xml version="1.0" encoding="utf-8"?>

<DATA>
 <ROW>
  <가족ID>28</가족ID>
  <신상ID>1</신상ID>
  <가족관계구분>배우자</가족관계구분>
  <성명>김 </성명>
  <주민등록번호> </주민등록번호>
  <학력구분>대졸</학력구분>
  <직업_학교명>초등학교교사</직업_학교명>
  <부양가족공제대상여부>1</부양가족공제대상여부>
  <의료보험대상여부>1</의료보험대상여부>
  <가족수당대상여부>1</가족수당대상여부>
  <생성일시>2025-09-19 14:36:26</생성일시>
  <수정일시>2025-09-19 14:36:26</수정일시>
</ROW>
<ROW>
  <가족ID>29</가족ID>
  <신상ID>1</신상ID>
  <가족관계구분>자녀</가족관계구분>
  <성명>박 </성명>
  <주민등록번호> </주민등록번호>
  <학력구분>유치원</학력구분>
  <직업_학교명>햇님유치원</직업_학교명>
  <부양가족공제대상여부>1</부양가족공제대상여부>
  <의료보험대상여부>1</의료보험대상여부>
  <가족수당대상여부>1</가족수당대상여부>
  <생성일시>2025-09-19 14:36:26</생성일시>
  <수정일시>2025-09-19 14:36:26</수정일시>
</ROW>
<ROW>
  <가족ID>30</가족ID>
  ....

content/table0/table0.xml

<그림 41> 행 기반 XML 구조 방식

② **LOB 데이터:** BLOB 또는 CLOB의 대용량 데이터는 '1) Base64 인코딩 방식'과 '2) lob 폴더 저장 방식' 중 하나로 보존포맷에 포함될 수 있으며, 1)번 방식이 기본(default) 방식임

1) **Base64 인코딩 방식:** '행 기반 XML 구조'로 LOB 데이터를 저장할 때, 해당 요소값을 Base64로 변환하여 기록하는 방식임. 이러한 변환 기능은 대부분의 DBMS 전용 애플리케이션에서 함수 형태로 제공됨. 예를 들어 MySQL에서는 TO_BASE64(컬럼명) 함수를, Oracle에서는 UTL_RAW.CAST_TO_RAW(컬럼명) 함수를 사용하여 Base64 값을 생성할 수 있음. <그림 42>는 '행 기반 XML 구조' 형식으로 작성된 table1.xml의 데이터 중 BLOB으로 선언되어 있는 '사진' 컬럼을 Base64 인코딩 방식으로 저장되는 예시를 보여줌. DBMS 차원에서 LOB 타입의 크기 제한이 있으며, 이는 이관 과정에서의 dump 파일 생성까지도 고려한 특성임. 따라서 LOB 데이터까지 하나의 파일로 처리해도 무리가 없다고 판단되며, 이에 따라 본 보존포맷은 Base64 인코딩 방식을 기본 저장 방식으로 채택함

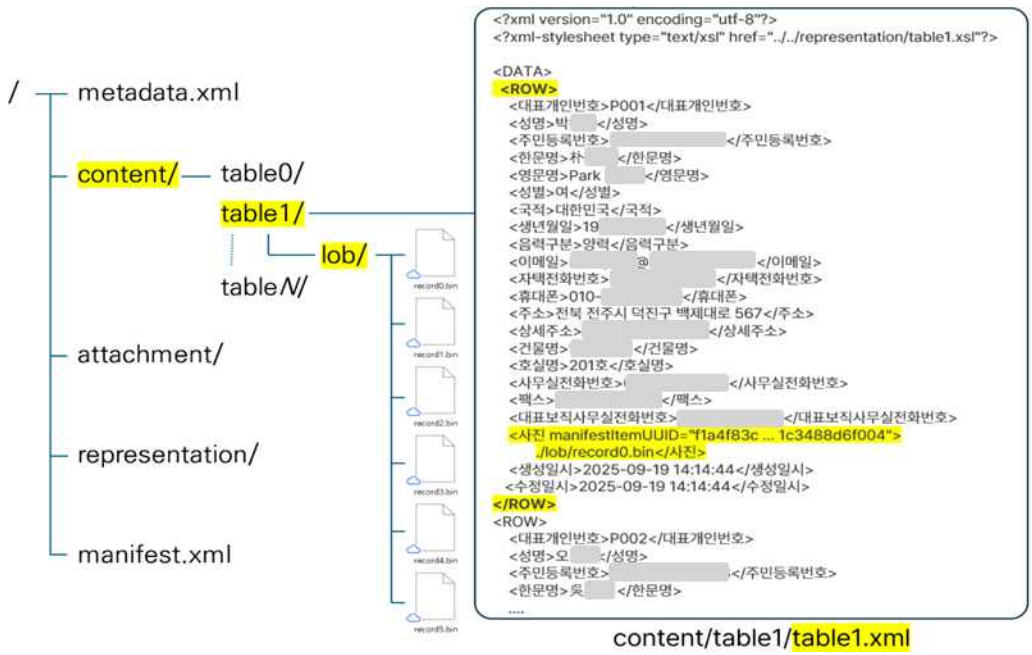
<?xml version="1.0" encoding="utf-8"?>

<DATA>
 <ROW>
  <대표개인번호>P001</대표개인번호>
  <성명>박 </성명>
  <주민등록번호> </주민등록번호>
  <한문명>朴 羅 美</한문명>
  <영문명>Park </영문명>
  <성별>여</성별>
  <국적>대한민국</국적>
  <생년월일> </생년월일>
  <음력구분>양력</음력구분>
  <이메일> @ </이메일>
  <자택전화번호> </자택전화번호>
  <휴대폰>010- </휴대폰>
  <주소>전북 전주시 덕진구 백제대로 567</주소>
  <상세주소> </상세주소>
  <건물명>인문2호관</건물명>
  <호실명>201호</호실명>
  <사무실전화번호> 1</사무실전화번호>
  <팩스> 1</팩스>
  <대표보직사무실전화번호>063-270-2001</대표보직사무실전화번호>
  <사진>UklGRiAjAABXRUJQVlA4ICAjdD4gR1dGKlBfICAgTyB6ICAg
  ...
  AAAgIAAgASpvASABPlEmlEAglH4glCAglCAgfiAMX2Ag</사진>
  <생성일시>2025-09-19 14:14:44</생성일시>
  <수정일시>2025-09-19 14:14:44</수정일시>
</ROW>
<ROW>
  <대표개인번호>P002</대표개인번호>
  <성명>오 </성명>
  <주민등록번호> </주민등록번호>
  <한문명>吳 </한문명>
  ....

content/table1/table1.xml

<그림 42> Base64 인코딩 방식

2) **lob 폴더 저장 방식:** 용량, Base64 인코딩 방식 미지원 등의 기술적인 사유로 '1) Base64 인코딩 방식'으로 정의되기 어려운 경우, LOB 데이터는 별도의 content/table*N*/lob/ 폴더에 저장될 수 있음. 이때, 파일명은 BLOB일 경우는 record*M*.bin 형식으로, CLOB의 경우는 record*M*.txt로 함. 이때, *M*은 해당 테이블 내에서 LOB 객체의 순번을 의미함. *M*은 LOB 객체의 식별을 위해 0부터 순차적으로 증가하는 방식으로 번호를 부여(예: 해당 BLOB 객체가 포함된 행의 생성 순서대로 번호 지정). 예를 들어, table1의 3번째 생성된 BLOB에 대한 파일명을 record2.bin로 명명함. Base64 인코딩 데이터가 저장되어 있던 필드에는, 해당 파일이 위치한 lob/ 폴더의 상대경로와 보존포맷 내 파일 관리를 위해 부여된 식별자(manifestItemUUID 속성)가 table*N*.xml에 함께 기록됨. 또한 manifestItemUUID (속성), 암호화 해시 알고리즘(속성), 생성 시간(속성)과 함께 각 파일의 해시값과 경로명은 manifest.xml에 저장됨. <그림 43>은 table1.xml과 lob/ 폴더가 어떤 관계를 갖는지를 보여줌. table1/ 폴더에는 table1.xml과 lob/ 폴더가 함께 존재함. table1.xml의 <사진> 태그는 원래 LOB 객체를 포함하고 있던 필드이며, 보존포맷에 저장될 때는 이 LOB 객체가 별도의 파일로 분리되어 저장되며, <사진> 요소에는 파일의 경로인 './lob/record0.bin'가 기록됨. 이 파일은 table1/ 하위의 lob/ 폴더에 위치함



<그림 43> BLOB 데이터의 lob 폴더 저장 방식 및 content/lob/ 폴더 구성요소 과의 관계

○ 세 번째 절차: attachment/ 생성

- **[붙임파일 개요]** HTML이나 웹프로그래밍 환경에서는 파일을 직접 열거나 하이퍼링크로 연결해 접근할 수 있음. 그러나 LOB 파일은 먼저 파일 형태로 추출해야만 접근이 가능하므로, 파일을 열기까지 여러 단계를 거쳐야 하는 번거로움이 있음. 따라서, 국내 DB들은 파일을 LOB 형태로 직접 DB에 저장하지 않고, 대부분 외부 저장소의 경로명을 DB에 기록하고 그 경로에 파일을 저장하는 방식을 많이 사용함. 이런 파일을 붙임 파일이라고 하는데, 이 파일들도 데이터세트에 포함되므로 보존포맷에서 함께 관리해야 함

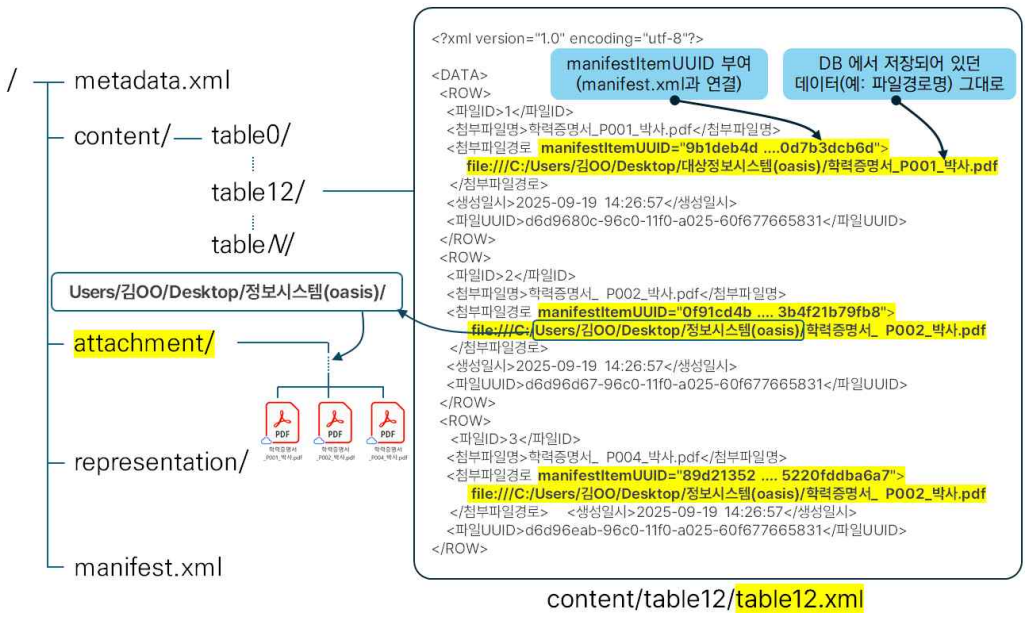
- **[별도 관리 필요성]** LOB는 데이터베이스 차원에서 크기 제한이 존재하므로 하나의 데이터 객체(예: dump 파일)로 묶어 저장할 수 있다고 가정할 수 있음. 반면 붙임파일은 크기 제한이 없어 대용량이 될 수 있으므로 table*N*.xml에 내재화하지 않고 보존포맷 내부에서 별도로 관리해야 함.

이를 위해 보존포맷 내부에 attachment/ 폴더를 두어, DB 필드에 기록된 외부 저장소의 경로나 주소를 참고해 실제 해당 위치에 있는 파일들을 모아 저장함. 또한 필요에 따라 붙임파일을 보존포맷 외부에 둘 수 있도록 허용해야 하므로, 이러한 특성을 수용할 수 있는 유연한 구조가 요구됨

- **[붙임 파일 경로 변경을 고려한 일원화 관리 방안]** 붙임 파일은 원래 DB 외부 저장소에 존재하고, DB에는 외부 저장소의 경로만이 기록되어 있음. 이런 붙임 파일을 보존포맷 내부로 이동 또는 다른 외부 저장소로 위치를 변경할 때마다 경로명이 달라짐. 만약 경로명이 table*N*.xml 안에 기록된다면, 위치가 바뀔 때마다 모든 관련 table*N*.xml을 일일이 찾아 수정해야 하는 문제가 발생함. 따라서 붙임 파일들의 위치가 바뀌더라도 table*N*.xml을 수정할 필요가 없도록 해야 함. 'manifest.xml'에서 경로 정보를 통합·관리하고, table*N*.xml 등의 파일에서는 붙임 파일을 manifestItemUUID와 같은 식별자를 통해 간접적으로 연계하는 방식이 적합함. 보존포맷 생성 시 최초 설정된 식별자만 유지하면 되므로, 파일 위치가 바뀌어도 table*N*.xml을 수정할 필요가 없음. <그림 44>는 붙임파일 관리를 manifest.xml에서 일원화하기 위해 manifestItemUUID로 연계하는 방식을 예시로 보여줌. 여기서 <첨부파일경로> 요소는 원래 DB에 저장되어 있던 파일 경로를 그대로 기록하는 컬럼에 해당하며, 이 요소에 manifestItemUUID 속성을 부여하여 manifest.xml에 정의된 UUID와 연결되도록 함. 이 방식은 해당 파일이 attachment/에 존재하더라도 table12.xml에는 경로 정보를 기록하지 않기 때문에, 이후 저장되는 위치가 변경되더라도 table12.xml을 수정할 필요가 없고 manifest.xml만 일괄 변경하면 됨

- **[attachment/ 폴더 내 붙임파일 경로 설정 방안]** attachment/의 파일은 manifest.xml에서 식별자로 관리되므로, 별도의 파일명 규칙을 둘 필요는 없음. 다만 파일 간 중복을 피하고 관리 편의성을 위해 가능한 한 원래 DB에 기록된 경로명을 활용함. <그림 44>에서 '학력증명서_P002_박사.pdf'의 원래 경로명은 C:/Users/김OO/Desktop/정보시스템(oasis)/학력증명서_P002_박사.pdf 임. 여기서 Windows 계열에서만 있는 드라이브 정보(C:)를 제외한 Users/김OO/Desktop/정보시스템(oasis)/학력증명서_P002_박사.pdf를 기준으로 attachment/Users/김OO/Desktop/정보시스템(oasis)/학력증명서_P002_박사.pdf 위치에 저장함

- **[붙임파일 존재 형태와 경로 설정]** <그림 44>의 예시에서는 붙임파일이 attachment/ 폴더에 개별 파일로 저장되어 있지만, 필요에 따라 동일 폴더 내에서 ZIP 파일로 묶이거나, 외부 저장소에 개별 또는 ZIP 형태로 존재할 수도 있음. 이러한 변경이 발생해도 tableN.xml에서는 manifestItemUUID로 해당 파일을 연계하므로 별도의 수정이 필요 없음. 또한 manifest.xml에서는 URI 방식과 ZIP 내부 경로 표기(./attachment/**attach_bundle.zip!**/cert/cert_A.hwp)를 모두 허용하므로, 파일이 외부 저장소에 있거나 ZIP으로 압축되어 있어도 문제없이 관리할 수 있음



<그림 44> 붙임 파일의 보존포맷 연계 방식(attachment/ 폴더에 위치)

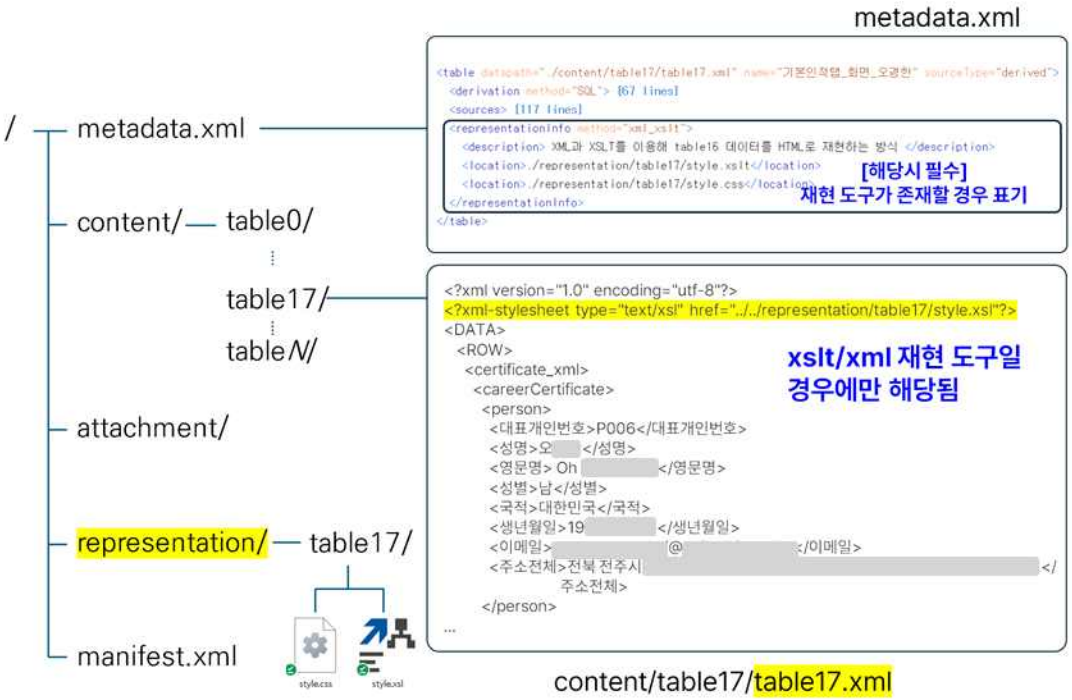
❍ 네 번째 절차: representation/ 생성

- **[representation/ 폴더 필요성]** DB 데이터는 단독으로 사용되기보다는, 이용자의 업무 편의를 위해 웹브라우저 기반의 GUI를 통해 제공되는 경우가 많음. 이러한 GUI를 활용하면 DB의 데이터를 기반으로 주요 기록물을 생성하고 열람할 수 있으며, 일부 데이터베이스는 이러한 기록물을 웹 환경에서 생성하기 위해 구축되기도 함. 따라서 이 경우에는 단순히 DB 데이터만 저장하는 것이 아니라, DB 데이터와 연계된 기록물을 재현할 수 있는 도구 또한 함께 관리·보존해야 함

- **[재현 도구와 해당 데이터 관계]** representation/ 폴더는 DB 데이터와 연계된 기록물을 재현하기 위한 도구를 저장하는 영역임. 이러한 재현 도구는 해당 데이터와 연결되어야 하므로, metadata.xml에서 해당 재현 도구에 연관된 데이터 정보를 담고 있는 <table> 요소의 <representationInfo> 하위 요소에 재현 도구의 경로와 관련 정보를 기록함

- **[재현 도구 구성 및 연결 방법]** 예를 들어 table16.xml이 XML/XSLT 방식으로 재현되는 경우, style.xslt와 style.css가 해당 테이블의 재현 도구가 됨. 이 두 파일은 representation/ 폴더에 저장하고, table16.xml과 연결된 <table> 요소의 <representationInfo>에 method 속성('xml_xslt'), <description> 요소(재현 도구 개요 설명), <location> 요소(재현 도구 위치 정보)를 <그림 45>와 같이 설정하면 됨. 파일명에 대한 별도 규칙은 없으나, 중복되지 않도록 관리하는 기본 원칙만 지키면 됨. 한 가지 제안으로, 재현 도구가 특정 <table> 요소에 종속되므로 해당 table*N*.xml의 번호를 활용해 <그림 45>와 같이 representation/ 안에 'table*N*' 폴더를 만들고 그 아래에 재현 도구를 배치하면 보다 체계적으로 관리할 수 있음

- **[재현 도구 파일의 manifest.xml 등록]** 재현 도구 파일들은 모두 보존포맷을 구성하는 파일들에 포함되므로 암호화 해시 알고리즘(hashAlgorithm 속성), 생성시간(createDateTime 속성), 식별자(manifestItemUUID 속성), 암호화 해시값(hashValue 요소), 경로명(location)은 manifest.xml에 저장됨. 만약 xslt/xml 방식으로 재현된다면, <그림 45>와 같이 관련 table*N*.xml 파일 내에도 xsl 파일의 위치가 명시될 것임



<그림 45> table*N*.xml과 metadata.xml, 그리고 representation/ 폴더 간의 관계와 저장 예시

❍ 다섯 번째 절차: manifest.xml 생성

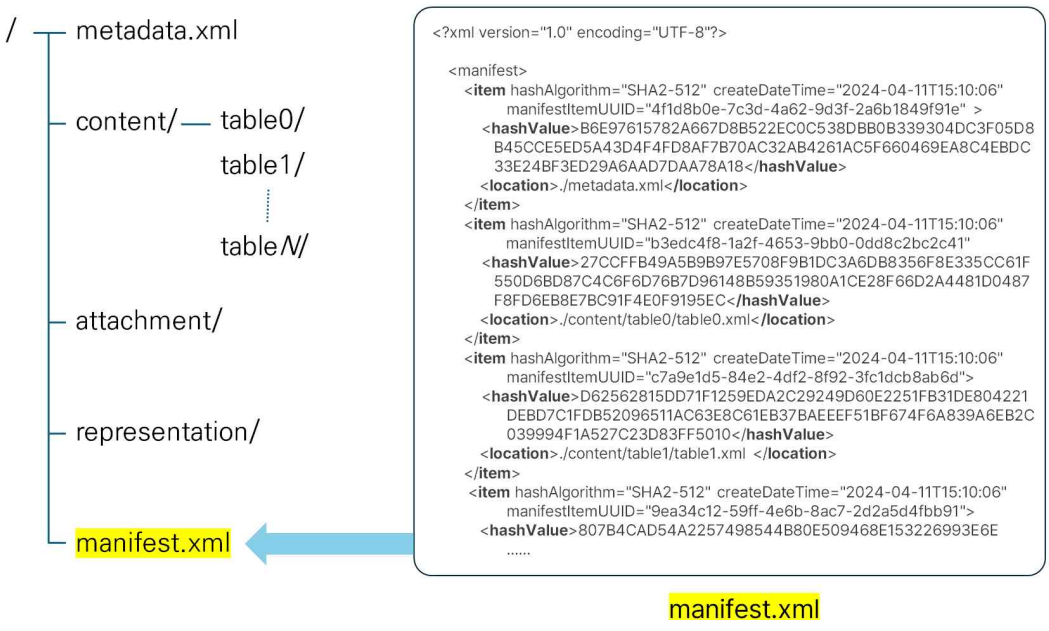
- **[manifest.xml 개요]** manifest.xml은 보존포맷 내 모든 파일을 식별하고 무결성을 검증하기 위한 중앙 관리 목록(central registry) 역할을 수행함
- **[경로정보의 단일 저장(일원화)]** 파일의 실제 위치는 manifest.xml에서만 관리하며, 다른 파일(table*N*.xml 등)에서는 manifestItemUUID를 기록함. 따라서 파일 위치가 변경되더라도 table*N*.xml 등의 내용을 수정할 필요가 없음
- **[내부·외부 파일 모두 지원]** attachment/ 내부에 있는 파일뿐 아니라 외부 저장소(URI), zip 내부 경로(zip:///...) 등의 다양한 위치 표현을 <location> 요소에서 허용함
- **[장기보존에 적합한 무결성 검증 체계]** 암호화 해시 알고리즘, 해시값을 기록함으로써 향후 보존 과정에서 파일 변조 가능성을 정확히 검증할 수 있음
- **[manifest.xml 구성]** 문서의 최상위 요소는 <manifest>이며, 그 하위에 하나 이상의 <item> 요소가 반복적으로 구성됨. 각 <item> 요소는 보존포맷에 포함된 개별 파일을 하나의 단위로 표현하며, 다음과 같은 속성과 하위 요소를 가짐

[속성(attribute)]

- manifestItemUUID: 보존포맷 내부에서 파일의 고유 식별을 위한 UUID. 붙임 파일과 LOB 파일이 Base64 인코딩 방식이 아닌, lob 폴더에 저장되는 경우 UUID를 통해 파일을 연계할 수 있음
- hashAlgorithm: 파일의 무결성을 검증하기 위해 사용한 암호화 해시 알고리즘(예: SHA2-512)
- createDateTime: manifest 항목의 생성 시점을 기록하며, 패키지 생성·이관 기록 관리에 활용됨

[하위 요소(child elements)]

- <HashValue> 지정된 해시 알고리즘으로 계산된 파일의 암호화 해시값을 저장함. 이 값은 장기보존 과정에서 파일 변조 여부를 검증하는 데 사용됨
- <Path> 보존포맷 기준 경로(relative path 또는 URI)로 기록된 실제 파일의 위치 정보. 외부 저장소에 저장된 붙임파일의 경우 URI 또는 zip 내부 경로 표기를 허용함
- <그림 46>은 manifest.xml의 예시를 보여줌



<그림 46> manifest.xml 예시

다. 오픈포맷 기반 보존포맷의 공공기관 실무 적용 및 정책적 지원 방안

❍ 오픈포맷 기반 보존포맷 설계의 원칙과 의의

- 본 연구과제에서 제안한 오픈포맷 기반 보존포맷은 기록의 4대 속성을 보장하고, 기존 SIARD 방식의 한계를 보완하며, 데이터 기반 활용성을 확보한다는 데이터세트(DB형) 보존포맷 설계 원칙에 따라 구축됨

- 그중에서도 가장 중요한 출발점은, 특정 전용 소프트웨어에 의존하지 않고 오픈소스 기반 응용프로그램만으로 보존포맷을 생성할 수 있어야 한다는 점임. 이는 공공기관 실무자가 고도의 전문지식이 없더라도 보존포맷을 이해하고 일부 절차를 수작업으로 수행하면서 전체 보존포맷을 조립할 수 있는 구조를 마련해야 함을 의미함

❍ 공공기관 실무 환경에서의 어려움과 요구

- 데이터세트 기록관리 체계는 공공기관에 갑작스럽게 도입되었고, 많은 기관에서 이를 새로운 업무 부담으로 인식하고 있음

- 실무자는 복잡한 데이터 구조나 보존포맷 기술을 충분히 이해하지 못하는 경우가 상당 부분 존재하며, 기록관리 교육과정에서도 데이터베이스 교육이 충분히 이루어지지 않고 있음. 기록연구사를 대상으로 한 재교육 또한 미흡하여, 데이터베이스 기반 기록관리 체계를 실무에서 바로 적용하기 어려운 상황임. 이러한 현실을 고려할 때, 공공기관이 스스로 보존포맷을 생성할 수 있도록 직관적이고 구현 가능한 방법론을 제시하는 것이 무엇보다 중요함

❍ 오픈포맷 기반 보존포맷의 실무 적용 가능성

- 본 연구에서 제안한 보존포맷은 전용 프로그램 없이도, 일부 절차는 수작업으로, 일부 절차는 오픈소스 기반 도구를 활용하여 생성할 수 있도록 설계됨. 시간이 지나면 스크립트나 간단한 자동화 도구를 활용하여 업무를 더 효율적으로 처리할 수 있는 가능성도 열려 있음. 특히 데이터베이스 교육을 한 학기 정도 이수한 경험이 있는 실무자라면 보존포맷의 구조를 이해하고 실제 생성 절차를 수행할 수 있을 만큼 난이도를 조정한 점이 중요한 특징임

❍ 안정적인 현장 적용을 위한 지원 체계의 필요성

- 그러나 데이터베이스 관련 실무 역량과 교육 인프라의 부족을 고려하면, 단순히 보존포맷을 제시하는 것만으로는 공공기관이 이를 안정적으로 적용하기 어려움. 따라서 보존포맷을 실무에 실제로 적용하기 위해서는 다음과 같은 지원 체계가 함께 마련되어야 함

❍ 보존포맷의 현장 적용을 위한 실천적 지원 방안

- 보존포맷은 원칙적으로 공공기관에서 직접 생성함. 이를 위해 국가기록원은 공공기관 실무자를 대상으로 정기적인 교육과 홍보를 실시하여 보존포맷 생성 역량을 강화함

- 공공기관은 기술 인력 부족, 데이터베이스 구조에 대한 이해 부족, 시스템 접근 제한, 보존포맷 생성 경험 부족 등 여러 여건과 환경적 제약으로 인해, 이관 대상 데이터세트를 자체적으로 보존포맷으로 생성하지 못하는 상황이 발생할 수 있음. 이러한 경우에는 해당 기관이 실제 데이터, 스키마 및 기타 요소, 첨부파일, 재현도구, 산출물 등을 각각 제출하도록 하고, 영구기록물관리기관이 이를 기반으로 보존포맷을 생성함

- 보존포맷 생성이 어려운 기관에 대해서는 집중적인 교육과 맞춤형 가이드 제공을 강화하여, 장기적으로 기관 스스로 보존포맷을 생성할 수 있는 역량을 확보하도록 지원함

3.5 데이터세트 유형에 적합한 장기보존패키지(NEO3) 규격 설계

가. 장기보존패키지 (NEO3) 구조 및 구성요소 정의

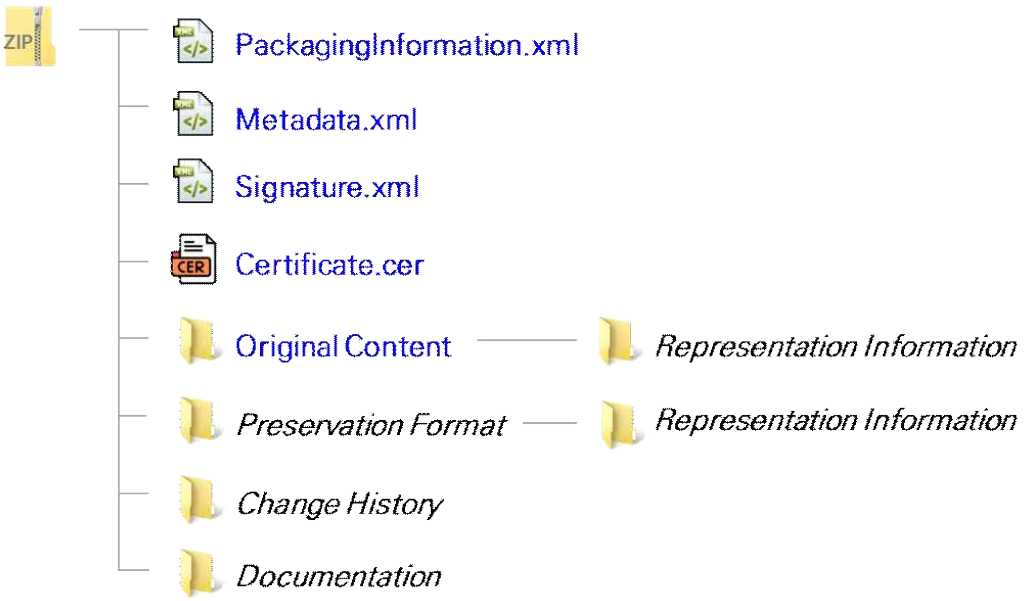
○ 장기보존패키지 구조 및 구성요소의 법·표준 근거

- 장기보존패키지의 법적 정의는 「공공기록물법 시행령」 제2조 제13호와 기록관리 공공표준(NAK 31-2)에 정의되어 있다. 두 개의 규정에 서로 다른 용어를 사용하고 있는 요소들도 있지만, 다음 <그림 47>과 같이 매핑되며 각각의 구성요소는 동일한 역할을 함

공공기록물관리법 시행령	장기보존패키지(NEO3) 공공표준 (NAK 31-2)
가. 전자기록물	가. 원문
나. 기록관리 메타데이터	나. 장기보존 메타데이터
다. 보존포맷	다. 보존포맷
라. 행정전자서명 등 진본확인 정보	라. 진본확인 정보
마. 그 밖에 전자기록물관리에 필요하다고 중앙기록물관리기관의 장이인정하는 정보	마. 장기보존 메타데이터 스키마 정보, 장기보존 메타데이터 스타일시트 (StyleSheet)

<그림 47> 장기보존패키지 구성요소 관련 법·표준

○ 장기보존패키지의 구조 개요: NEO3 패키지는 ZIP 컨테이너를 기본으로 하며 다음 <그림 48>과 같은 폴더구조와 파일들을 구성요소로 하며, 각 구성요소의 내용은 다음과 같음. 파란색은 필수, 검은색/기울임은 조건부 필수(해당 정보가 존재할 경우)임



<그림 48> 장기보존패키지 구성요소 관련 법·표준

- PackagingInformation.xml [필수]: 패키지 버전, 개요, 구성요소 목록, 각 구성요소의 해시값(SHA-계열) 등 패키징 정보임. 패키지 버전, 디렉터리·파일 인벤토리, 알고리즘, 해시값을 포함

- Metadata.xml [필수]: 기록관리 메타데이터(PDI 포함). 원문·보존포맷 파일에 대한 해시값 참조를 명시함. 기록 식별·발견·이해 지원을 위한 요소 세트와 Preservation Description Information (예: 출처, 고유 식별자, 무결성·권리 정보 등)을 포함함. 원문·보존포맷 파일의 해시값을 수록하여 패키지 내 참조 일관성을 확보함

- Signature.xml [필수] & Certificate.cer [필수]: 행정전자서명 정보. Metadata.xml에 대한 전자서명을 통해 원문·메타데이터·보존포맷 무결성 검증을 간접적으로 보장함. Metadata.xml에 대한 행정전자서명 적용. 이를 통해 메타데이터－원문－보존포맷 간 무결성 체인을 형성함

- Original Content/ [필수]: 원문(예: 텍스트형 덤프 등)으로 생산 당시의 원문. 파일별 해시를 PackagingInformation.xml과 Metadata.xml에 상호 참조함

- Preservation Format/ [조건부 필수(보존포맷 변환이 발생한 경우)]: 장기보존을 위한 변환 결과(표준 개방형 우선)로 장기열람 보장을 위한 변환본. 형식 선정 시 공개성, 문서화, 생태계 성숙도, 검증 도구 가용성을 고려함

- Representation Information/ [조건부 필수(재현도구 구성요소가 별도로 존재하는 경우)]: 재현정보를 별도 폴더로 관리할 필요가 있을 때 사용함

- Change History/ [조건부 필수(변경 이력이 발생한 경우)]: 변경 이력 관리(프로젝트 정책에 따라 운용)를 위한 구성요소임

- Documentation/ [필수]: 재현 보조 산출물·스크립트 등(해당 보존포맷이 수용 가능한 형식)으로 재현에 필요한 부가 산출물(뷰어 스크립트, 매핑 정의 등)을 포함하되, 보존포맷이 수용 가능한 범위로 제한함

나. 장기보존패키지가 보존해야 할 데이터세트 유형 기록물의 범위 정의

○ 데이터세트 유형 전자기록물의 보존 범위를 정의함. 보존 범위는 필수 포함(필수), 조건부 포함(선택), 이해 보조 목적의 선택 포함(선택·맥락)으로 구분함. 이 구분은 패키지 설계·수집·검증 단계의 판단 기준으로 사용함. 데이터세트 유형 기록물의 보존대상은 ‘실제 데이터’, ‘붙임 파일’, ‘스키마 및 기타요소’, ‘재현 정보’, 그리고 ‘주요 산출물’임

○ 필수 포함: DB의 실제 데이터(테이블, 레코드 등 데이터 자체)와 외부 저장소의 붙임 파일(예: 회의록 첨부파일, 사진·증빙 등 원문 첨부)은 ‘필수’적으로 포함되어야 함

○ 조건부 포함(선택): 스키마 구조, 제약조건, 트리거, 프로시저, 뷰 등 DB의 구조와 동작을 정의하는 요소임. 이러한 요소들은 DBMS 기능이 기록 재현에 필요하다고 판단되는 경우에 ‘선택’적으로 포함. 특히 데이터의 구조적 의미나 재현 과정에 직접적으로 영향을 미치는 요소들은 보존 대상으로 포함할 수 있으며, 불필요한 경우에는 제외할 수 있음

○ 조건부 포함(선택): 재현정보(이용자 기능/화면): 데이터가 특정 기능·화면 구동을 전제로 만들어졌거나, 기능 맥락 없이는 의미가 훼손되는 경우 ‘선택’적으로 포함되어야 함

○ 이해 보조 목적의 선택 포함(선택·맥락): 주요 산출물(제안요청서, 용역설계서, 시스템구성도, ERD, 컬럼정의서, 테이블명세서, 사용자·운영자 매뉴얼, 메뉴구조도, 화면설계서 등): 전자기록물 범위에는 직접 포함되지 않지만, 내용 이해를 위한 직접적 정보이므로 존재 시 ‘선택·맥락’적으로 포함되어야 함

다. 데이터세트 유형에 적합한 '장기보존 메타데이터' 설계

❍ 데이터세트 유형 NEO3 장기보존 메타데이터 개요

- 데이터세트 유형 NEO3 장기보존 메타데이터(안)는 국가기록원에서 제시한 SIP 메타정보와 전북대 연구팀이 제안한 데이터세트 메타데이터(안)를 조합하여 도출함

· 장기보존 메타데이터 구조는 국가기록원의 SIP 구조를 기반으로 하되, 일부 요소의 위치를 조정하여 계층을 재구성함

· 국가기록원과 전북대 연구팀에서 공통으로 제시한 요소를 핵심 메타데이터로 유지하고, 데이터세트의 특성에 맞춰 일부 요소를 추가·수정·삭제함

- 데이터세트 유형 NEO3 장기보존 메타데이터는 총 4계층으로 이루어짐

· 6개 차상위요소, 29개 하위요소, 52개 최하위요소, 36개 세부요소로 구성됨

· 차상위요소는 객체메타데이터, 생산자, 기록관리식별자, 기록물명, 시스템정보, 데이터세트정보로 구성되며, 기록관리식별자, 기록물명을 제외한 다른 요소는 각각 하위요소를 가짐

❍ 객체메타데이터 차상위요소

- 객체메타데이터(ObjectCon)는 장기보존패키지 객체에 대한 기본정보를 기술하는 요소이며, <표 23>과 같이 5개 하위요소로 구성됨

<표 23> 객체메타데이터(ObjectCon) 차상위요소의 구성

차상위요소								
요소명	내용					필수여부	반복여부	
객체메타데이터	장기보존패키지 객체의 기본 정보					필수	-	
하위요소			최하위요소			세부요소		
요소명	필수 여부	반복 여부	요소명	필수 여부	반복 여부	요소명	필수 여부	반복 여부
객체유형	필수	-						
객체버전	필수	-						
객체유형설명	필수	-						
객체생성일시	필수	-						
패키징범위	필수	-						

- 객체유형(ObjectType)

· 장기보존패키지의 유형 정보를 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함

· 【데이터세트 장기보존패키지】, 【수정된 데이터세트 장기보존패키지】 중 택1

· 【데이터세트 장기보존패키지】: 최초 생성된 패키지를 뜻함

· 【수정된 데이터세트 장기보존패키지】: 기존 패키지에 메타데이터 변경, 보존포맷 변환 등의 변경사항이 발생할 경우, 이를 반영하여 새롭게 생성된 패키지를 뜻함

- 객체버전(ObjectVersion) 하위요소

· 장기보존패키지 버전(표준명)을 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함

· 【NAK 31-1:2022(v(2.3)】, 【NAK 31-2:2022(v1.1)】, 【NAK/TS 3:2008(v1.0)】, 【NAK 31:2013(v2.0)】, 【NAK 31:2017(v2.1)】, 【NAK 31-1:2020(v2.2)】 중 택1

- 객체유형설명(ObjectTypeDescription)

· 장기보존패키지 유형에 대한 설명을 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함

· '객체유형' 하위요소의 값이 무엇이냐에 따라 선택값이 결정됨

· '객체유형' 요소의 값이 【데이터세트 장기보존패키지】일 경우: 【행정정보시스템에서 생산된 데이터세트를 기반으로 최초 생성된 장기보존패키지】를 선택함

· '객체유형' 요소의 값이 【수정된 데이터세트 장기보존패키지】일 경우: 【기존 장기보존패키지에 수정사항(메타데이터, 컴포넌트, 보존포맷 등 변경)이 발생할 경우, 이를 반영한 장기보존패키지】를 선택함

- 객체생성일시(ObjectCreationDate)

· 장기보존패키지(SIP) 생성일시를 기술하는 요소로, 해당 요소의 값은 시스템에 의해 자동으로 입력되도록 설계함

· 날짜와 시간을 일관된 표준 형식에 맞춰 기술하도록 함(예: 2025-09-05T14:45:01)

- 패키징범위(PackagingScope)

· 장기보존패키지에 포함된 데이터세트의 범위(전체 또는 일부)를 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함

· 【시스템 전체】, 【시스템 일부 단위기능】 중 택1

· 【시스템 전체】: 시스템 내 모든 데이터세트(단위기능)를 포함한 패키지를 뜻함

· 【시스템 일부 단위기능】: 시스템 내 일부 데이터세트(단위기능)만 포함한 패키지를 뜻함

○ 생산자 차상위요소

- 생산자(CreatorCon)는 장기보존패키지 내 데이터세트를 생산한 기관에 대한 정보를 기술하는 요소로, 다음 <표 24>와 같이 6개 하위요소로 구성됨

· 데이터세트는 특정 행정정보시스템을 통해 생산된 기록으로, 생산자를 개인 단위로 특정하기 어렵다는 한계가 있음

· 행정정보시스템에는 다수의 사용자가 입력한 데이터가 테이블(행과 열) 형태로 저장되어 있음. 데이터세트에 철-건-컴포넌트 계층을 적용하면 테이블의 각 레코드(행)를 기록물건으로 간주해야 하며 레코드별로 생산자를 입력해야 함

· 이 경우 데이터세트 안에 여러 생산자가 포함될 수 있음. 이러한 복잡성을 줄이기 위해, 개별 사용자 대신 해당 데이터세트를 운영·관리하는 기관을 기재하는 것을 원칙으로 하되, 필요한 경우 선택적으로 부서명이나 개인명을 추가 기재할 수 있도록 함

- 생산자 요소는 해당하는 생산 주체의 수만큼 반복하여 작성할 수 있음

· 데이터세트 보유기관과 시스템 관리기관이 단일기관으로 동일한 경우에는 생산자 요소를 반복하여 기술하지 않음

· 데이터세트 보유기관과 시스템 관리기관이 다르거나 보유기관이 여러 곳인 경우, 해당하는 모든 주체를 개별적으로 식별하여 생산자 요소를 반복 기술하도록 함

<표 24> 생산자(CreatorCon) 차상위요소의 구성

차상위요소								
요소명	내용					필수여부	반복여부	
생산자	장기보존패키지 내 데이터세트의 생산자 정보					필수	반복	
하위요소			**최하위요소**			**세부요소**		
요소명	필수 여부	반복 여부	요소명	필수 여부	반복 여부	요소명	필수 여부	반복 여부
생산자유형	필수	-						
기관명	필수	-						
기관코드	선택	-						
부서명	선택	-						
부서코드	선택	-						
개인명	선택	-						

- 생산자유형(CreatorType)

· 데이터세트 생산자 유형 정보를 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함

· 【데이터보유기관】, 【시스템관리기관】, 【기타】 중 택1

· 데이터세트 보유기관과 시스템 관리기관이 단일기관으로 동일한 경우에는 【데이터 보유기관】 값 하나만 선택하도록 함

- 기관명(CorporateName)

· 데이터세트 생산자가 속한 기관명을 기술하는 요소

· 기관명은 약칭이나 중앙행정기관, 공공기관 등과 같은 일반적 표현이 아닌, 해당 기관의 법적·행정적으로 등록된 정식 명칭을 정확히 입력해야 함

- 기관코드(CorporateCode)

· 데이터세트 생산자가 속한 기관의 식별코드를 기술하는 요소

· 행정표준코드관리시스템(http://www.code.go.kr)의 기관코드를 참고하여 작성함

- 부서명(SectionName)

· 데이터세트 생산자가 속한 부서명을 기술하는 요소

· 특정 부서가 데이터세트의 생산 주체로 식별될 수 있는 경우에 선택적으로 입력 가능함

- 부서코드(SectionCode)

· 데이터세트 생산자가 속한 부서코드를 기술하는 요소

· 특정 부서가 데이터세트의 생산 주체로 식별될 수 있는 경우에 선택적으로 입력 가능함

- 개인명(PersonName)

· 데이터세트 생산자 개인의 이름을 기술하는 요소

· 생산 주체가 특정 개인으로 특정될 수 있는 경우에 선택적으로 입력 가능함

○ 기록관리식별자 차상위요소

- 기록관리식별자(RecordManagementID)는 데이터세트 장기보존패키지의 식별코드를 기술하는 요소로, 별도의 하위 항목을 갖지 않음

- 기록관리식별자는 ① 관리의 기본단위, ② 기록관리, ③ 정보 제공·처리의 측면을 고려해야 함

  ① 관리의 기본단위 측면: 기록관리식별자에는 행정정보 데이터세트가 '시스템 단위'로 관리된다는 규정에 따라 정보시스템 식별코드를 포함해야 함

  ② 기록관리 측면: 여러 데이터세트와 다양한 방식의 장기보존패키지를 구분하기 위해 기록관리식별자에는 '순차적 일련번호'가 필요함

  ③ 정보 제공·처리 측면: 기록관리식별자는 '검색·정렬·인덱싱'을 위해 기록물 유형, 생산연도, 이관연도, 기관 정보 등 직관적 정보를 포함해야 함

- (식별코드 체계 제안) 기록물 유형(3)+기관코드(7)+행정정보시스템식별코드(10)+일련번호(10~)

· 기록물유형(3자리): DOC, DET 등 행정정보 데이터세트는 DAT로 사용

· 기관코드(7자리): 행정표준코드관리시스템(http://www.code.go.kr)의 기관코드

· 행정정보시스템식별코드(10자리): 정보자원관리시스템(IRM, http://www.irm.go.kr)에서 확인할 수 있는 식별코드 (※ IRM을 사용하지 않는 기관이나 IRM에 등록되지 않은 시스템일 경우, 협의를 통한 보완책 마련 필요)

· 일련번호(10자리 이상): 동일한 행정정보시스템 및 데이터세트 범주 내에서 발생하는 중복이나 변동을 식별하기 위하여 부여하는 보조 식별번호

<표 25> 기록관리식별자(RecordManagementID) 차상위요소의 구성

차상위요소			
요소명	내용	필수여부	반복여부
기록관리식별자	데이터세트 장기보존패키지의 식별코드	필수	-

○ 기록물명 차상위요소

- 기록물명(RecordTitle)은 장기보존패키지 생산 시 부여된 제목을 기술하는 요소로, 별도의 하위 항목을 갖지 않음

<표 26> 기록물명(RecordTitle) 차상위요소의 구성

차상위요소			
요소명	내용	필수여부	반복여부
기록물명	장기보존패키지에 부여된 제목	필수	-

○ 시스템정보 차상위요소

- 시스템정보(SystemInfoCon)는 데이터세트를 생산 및 관리하는데 사용되는 행정정보시스템에 대한 정보를 기술하는 요소로, 다음 <표 27>과 같이 8개 하위요소, 15개 최하위요소, 4개 세부요소로 구성됨

· 데이터세트의 장기보존과 지속적 활용을 위해 행정정보시스템의 구성 및 산출물 정보를 메타데이터로 관리해야 함. 이는 데이터세트가 생산된 원래의 기술환경과 업무 맥락에서 분리될 경우 데이터의 구조와 의미를 정확히 해석하기 어려워지기 때문임

<표 27> 시스템정보(SystemInfoCon) 차상위요소의 구성

차상위요소								
요소명	내용					필수여부	반복여부	
시스템정보	데이터세트를 생산 및 관리하는데 사용되는 행정정보시스템 정보					필수	-	
하위요소			**최하위요소**			**세부요소**		
요소명	필수여부	반복여부	요소명	필수여부	반복여부	요소명	필수여부	반복여부
시스템명	필수	-						
시스템식별자	필수	-						
시스템개요	선택	-						
시스템구축연도	필수	-						
시스템폐기연도	해당시 필수	-						
DBMS정보	필수	반복	DBMS명/버전	필수	-			
			업무DB명	필수	-			
연계시스템	해당시 필수	반복	연계유형	해당시 필수	-			
			연계시스템명	해당시 필수	-			
			연계시스템 식별자	해당시 필수	-			
			연계시스템 보유기관명	해당시 필수	-			
			연계시스템 보유기관코드	해당시 필수	-			
			연계형태	선택	-	연계방식	선택	-
						연계방향	선택	-
산출물	해당시 필수	반복	산출물유형	필수	-			
			산출물제목	필수	-			
			산출물포맷	필수	-	포맷명	필수	-
						포맷버전	선택	-
			산출물크기	필수	-			
			산출물위치	필수	-			
			ManifestID참조	필수	-			
			산출물CI식별자	필수	-			

- 시스템명(SystemName)

· 행정정보시스템의 공식명칭(통용명칭)을 기술하는 요소

· 데이터세트 관리기준표의 '시스템명' 요소에 기재된 정보를 참조하여 작성함

- 시스템식별자(SystemID)

· 행정정보시스템의 식별코드를 기술하는 요소

· 정보자원관리시스템(IRM, http://irm.go.kr)에서 관리하는 시스템 식별코드를 사용하되, IRM을 사용하지 않는 기관이거나 IRM에 등록되지 않은 시스템의 경우, 기관 내에서 자체적으로 부여·관리하는 유일한 식별코드를 작성함

- 시스템개요(SystemOverview)

· 행정정보시스템의 목적, 주요 기능 등 시스템 전반에 관한 개요를 기술하는 요소

· 데이터세트 관리기준표의 '시스템 개요' 요소에 기재된 정보를 참조하여 작성함

- 시스템구축연도(SystemImplementationYear)

· 행정정보시스템의 공식 운영 개시 연도를 기술하는 요소

· 데이터세트 관리기준표의 '구축년도' 항목에 기재된 정보를 참조하여 작성함

- 시스템폐기연도(SystemDecommissioningYear)

· 행정정보시스템이 공식적으로 폐기된 연도를 기술하는 요소

· 폐기연도는 해당 시스템에서 생산된 모든 데이터가 더 이상 변경되지 않는 고정된 시점으로, 데이터의 생애주기가 보존 및 처분 단계로 전환되는 기준점이 됨

- DBMS정보(DBMSInfoCon)

· 행정정보시스템에서 사용하는 데이터베이스 관리 시스템(DBMS) 정보를 기술하는 요소로, 다음 <표 28>과 같이 2개의 최하위요소로 구성됨

<표 28> DBMS정보(DBMSInfoCon) 하위요소의 구성

최하위요소	내용
DBMS명/버전 (DBMSNameVersion)	· 행정정보시스템에서 사용 중인 데이터베이스 관리 시스템(DBMS)의 종류와 버전 정보를 기술하는 요소 · 행정정보 데이터세트 관리기준표의 'DBMS정보' 요소에 기재된 정보를 참조하여 작성함
업무DB명 (FunctionDBName)	· 행정정보시스템을 통해 생산된 데이터가 저장 및 관리되는 DB명을 기술하는 정보 · 행정정보 데이터세트 관리기준표의 '업무DB명' 요소에 기재된 정보를 참조하여 작성함

- 연계시스템(InterfacedSystemCon)

· 행정정보시스템과 연계되어 데이터 교환 및 처리를 수행하는 타 시스템 정보를 기술하는 요소로, 다음 <표 29>와 같이 6개 최하위요소와 2개 세부요소로 구성됨

<표 29> 연계시스템(InterfacedSystemCon) 하위요소의 구성

최하위요소	내용		
연계유형 (InterfaceType)	· 행정정보시스템과 타 시스템 간의 연계유형을 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함 · 【내부연계】, 【외부연계】, 【기타】 중 택1		
연계시스템명 (InterfacedSystem Name)	· 행정정보시스템과 연계된 타 시스템명을 기술하는 요소		
연계시스템식별자 (InterfacedSytemID)	· 행정정보시스템과 연계된 타 시스템의 식별코드를 기술하는 요소		
연계시스템 보유기관명 (InterfacedSystem CoporateName)	· 행정정보시스템과 연계된 타 시스템을 보유한 기관명을 기술하는 요소 · 시스템과 내부적 또는 외부적으로 연계되는 시스템명을 보유·관리하고 있는 기관명을 선택적으로 입력함		
연계시스템 보유기관코드 (InterfacedSystem CoporateID)	· 행정정보시스템과 연계된 타 시스템을 보유한 기관코드를 기술하는 요소 · 시스템과 내부적 또는 외부적으로 연계되는 시스템명을 보유·관리하고 있는 기관코드를 선택적으로 입력함		
연계형태 (InterfaceType)	· 행정정보시스템과 타 시스템 간 연계방식 정보를 기술하는 요소로, 2개 세부요소로 구성됨		
	세부요소	연계방식 (InterfaceMethod)	· 행정정보시스템과 연계된 타 시스템 간 데이터가 교환되는 기술방식을 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함 · 【EAI】, 【ESB】, 【Socket】, 【API】, 【DBLink】, 【JDBC】, 【기타】 중 택1
	세부요소	연계방향 (InterfaceDirection)	· 행정정보시스템과 연계된 시스템 간 데이터 흐름 방향을 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함 · 【송신】, 【수신】, 【송수신】 중 택1

- 산출물(DeliverableCon)

· 행정정보시스템의 최종 산출물 정보를 기술하는 요소로, <표 30>과 같이 7개 최하위요소와 2개 세부요소로 구성됨

· 시스템 산출물은 시스템 구축 및 운영 과정에서 생산된 기술 문서(시스템설계서, 이용자매뉴얼, 메뉴구조도, 테이블정의서 등)로, 데이터의 구조, 시스템 기능 등에 대한 정보를 담고 있음. 이는 데이터의 해석과 재현에 활용됨

· 행정정보시스템 산출물의 수만큼 '산출물' 요소를 반복하여 작성할 수 있음

<표 30> 산출물(DeliverableCon) 하위요소의 구성

최하위요소	내용		
산출물유형 (DeliverableType)	· 행정정보시스템 최종 산출물 유형을 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함 · 【제안요청서】, 【용역설계서】, 【시스템구성도】, 【메뉴구조도】, 【ERD】, 【컬럼정의서】, 【테이블정의서】, 【사용자매뉴얼】, 【운영자매뉴얼】, 【UI설계서】, 【연계시스템정의서】, 【기타】 중 택1		
산출물제목 (DeliverableTitle)	· 행정정보시스템 최종 산출물 제목을 기술하는 요소		
산출물포맷 (DeliverableFormatCon)	· 행정정보시스템 최종 산출물 파일포맷 정보를 기술하는 요소로, 2개 세부요소로 구성됨		
	세부요소	포맷명 (FormatName)	· 최종 산출물의 파일포맷명을 기술하는 요소
	세부요소	포맷버전 (FormatVersion)	· 최종 산출물의 파일포맷 버전을 기술하는 요소
산출물크기 (DeliverableSize)	· 행정정보시스템 최종 산출물의 파일 크기를 기술하는 요소 · Byte 단위로 작성함 (예. 2.5MB → 2,621,440Byte)		
산출물위치 (DeliverableLocation)	· 행정정보시스템 최종 산출물이 저장된 위치를 기술하는 요소 · URN(UUID, DOI 등), URL(path(파일경로), ftp, http) 등을 작성함		
ManifestID참조 (ManifestIDRef)	· 행정정보시스템 최종 산출물과 관련된 장기보존패키지의 ManifestID 속성값을 기술하는 요소 · 행정정보시스템 최종 산출물이 장기보존패키지 내에서 실제 어떤 파일과 대응되는지를 식별하기 위해 PackagingInformation.xml의 ManifestItem에 부여된 ManifestItemID 속성값을 참조하여 작성함		
산출물CI식별자 (DeliverableCIID)	· 행정정보시스템 최종 산출물과 관련된 장기보존패키지의 ContentID 속성값을 기술하는 요소 · 행정정보시스템 최종 산출물이 장기보존패키지 내 어떤 기록물과 대응되는지를 식별하기 위해 ContentInformation의 ContentItem에 부여된 ContentID 속성값을 참조하여 작성함 · 다만, 산출물이 기록물 정의에 포함되지 않을 때는 메타데이터는 작성 대상에서 제외함		

○ 데이터세트정보 차상위 요소

- 데이터세트정보(DatasetInfoCon)는 장기보존패키지에 포함된 데이터세트(단위기능)의 기록관리에 필요한 정보를 기술하는 요소로, <표 31>과 같이 10개 하위요소, 37개 최하위요소, 32개 세부요소로 구성됨

- 하나의 패키지에 여러 데이터세트(단위기능)가 포함될 경우 그 개수만큼 반복하여 작성할 수 있음

<표 31> 데이터세트정보(DatasetInfoCon) 차상위요소의 구성

차상위요소			
요소명	내용	필수여부	반복여부
시스템정보	데이터세트를 생산 및 관리하는데 사용되는 행정정보시스템 정보	필수	반복

하위요소			최하위요소			세부요소		
요소명	필수 여부	반복 여부	요소명	필수 여부	반복 여부	요소명	필수 여부	반복 여부
단위기능정보	필수	-	단위기능명	필수	-			
			단위기능식별자	선택	-			
			단위기능개요	필수	-			
			주제어	선택	-			
단위기능범위	필수	반복	DB질의문	선택	-			
			적용범위	필수	-			
			적용범위관련정보	선택	-			
권한	필수	반복	데이터보유권한	필수	반복	기관명	필수	-
						기관코드	선택	-
						부서명	선택	-
						부서코드	선택	-
			시스템관리권한	필수	반복	기관명	필수	-
						기관코드	선택	-
						부서명	선택	-
						부서코드	선택	-
			접근권한	필수	반복			
			비밀	해당시 필수	-	비밀분류	필수	-
						비밀분류근거	필수	-
						보호기간	필수	-
			공개	필수	반복	공개구분	필수	-
						비공개사유	해당시 필수	-
						공개제한부분	해당시 필수	-
						공개대상계층	해당시 필수	-
						공개대상범위	해당시 필수	-
생산기간	필수	-	생산일시	필수	-			
			종료일시	해당시 필수	-			
보존기간정보	필수	-	보존기간	필수	-			
			보존기간책정사유	필수	-			
			기산일	필수	-			
처분	선택	-	처분제약사항	선택	-			
			처분방법	선택	-			

매체	해당시 필수	반복						
관리이력	해당시 필수	반복	관리유형	필수	-			
			관리설명	해당시 필수	-			
			관리일시	필수	-			
			관리행위자	필수	-	기관명	필수	-
						기관코드	해당시 필수	-
						부서명	해당시 필수	-
						부서코드	해당시 필수	-
						개인명	필수	-
						직위(직급)명	선택	-
			변경요소	해당시 필수	반복	변경요소명	필수	-
						변경이전값	필수	-
보존이력	해당시 필수	반복	보존처리유형	필수	-			
			보존처리설명	해당시 필수	-			
			보존처리일시	필수				
			보존행위자	필수		기관명	필수	-
						기관코드	해당시 필수	-
						부서명	해당시 필수	-
						부서코드	해당시 필수	-
						개인명	필수	-
						직위(직급)명	선택	-
컴포넌트	해당시 필수	반복	컴포넌트유형	필수	-			
			컴포넌트제목	필수	-			
			컴포넌트개요	선택	-			
			컴포넌트포맷	필수	-	포맷명	필수	-
						포맷버전	선택	-
			컴포넌트크기	필수	-			
			컴포넌트위치	필수	-			
			컴포넌트세부위치	선택				
			컴포넌트CI식별자	필수	-			
			ManifestItemID참조	필수	-			

- 단위기능정보(UnitFunctionCon)

· 데이터세트의 관리단위(단위기능)에 대한 개요 정보를 기술하는 요소로, <표 32>와 같이 4개 최하위요소로 구성됨

<표 32> 단위기능정보(UnitFunctionCon) 하위요소의 구성

최하위요소	내용
단위기능식별자 (UnitFunctionID)	· 단위기능의 식별코드를 기술하는 요소
단위기능명 (UnitFunctionName)	· 데이터세트 기록관리 대상이 되는 단위기능명을 기술하는 요소 · 데이터세트 관리기준표의 '단위기능명' 요소에 기재된 정보를 참조하여 작성함
단위기능개요 (UnitFunctionOverview)	· 단위기능의 목적, 수행 범위 및 주요 업무 내용 등에 대한 정보를 기술하는 요소 · 데이터세트 관리기준표의 '단위기능개요' 요소에 기재된 정보를 참조하여 작성함
주제어 (SubjectWords)	· 단위기능과 관련된 주제어를 기술하는 요소 · 데이터세트 관리기준표의 '주제어' 요소에 기재된 정보를 참조하여 작성함

- 단위기능범위(UnitFunctionScopeCon)

· 단위기능이 포괄하는 데이터의 범위 정보를 기술하는 요소로, <표 33>과 같이 3개 최하위요소로 구성됨

· 데이터세트가 어떤 테이블, 컬럼 등을 포함하는지 작성함

<표 33> 단위기능범위(UnitFunctionScopeCon) 하위요소의 구성

최하위요소	내용
조회용쿼리 (RetrievalQuery)	· 단위기능에 해당하는 데이터를 추출하기 위해 사용된 DB 질의문을 기술하는 요소
적용범위 (ApplicationScope)	· 단위기능이 적용되는 DB 내 테이블 또는 데이터 영역 범위를 기술하는 요소 · 데이터세트 관리기준표의 '적용범위' 요소에 기재된 정보를 참조하여 작성함
적용범위관련정보 (ApplicationScopeInfo)	· 적용범위가 일부 테이블로 한정될 경우 관련 테이블을 기술하는 요소 · 데이터세트 관리기준표의 '적용범위 관련 정보' 요소에 기재된 정보를 참조하여 작성함 · 다만, '적용범위'가 단위기능 전체일 경우 별도의 테이블 구분이 필요하지 않음

- 권한(RightsCon)

· 장기보존패키지 내 데이터세트의 이용 및 접근을 관리하고 통제하기 위한 권한 정보를 기술하는 요소로, <표 34>와 같이 5개 최하위요소, 16개 세부요소로 구성됨

<표 34> 권한(RightsCon) 하위요소의 구성

최하위요소	내용		
데이터보유권한 (DataOwing CorporateCon)	· 행정정보시스템에서 생산·처리된 데이터세트를 공식적으로 소유 및 관리할 수 있는 권한을 가진 기관을 기술하는 정보로, 4개 세부요소로 구성됨 · 데이터세트 관리기준표의 '데이터 보유권한' 요소에 기재된 정보를 참조하여 작성함		
	세부요소	기관명 (CorporateName)	· 데이터세트에 대한 보유권한을 가진 기관명을 기술하는 요소
		기관코드 (CorporateCode)	· 데이터세트에 대한 보유권한을 가진 기관코드를 기술하는 요소
		부서명 (SectionName)	· 데이터세트에 대한 보유권한을 가진 부서명을 기술하는 요소
		부서코드 (SectionCode)	· 데이터세트에 대한 보유권한을 가진 부서코드를 기술하는 요소
시스템관리권한 (SystemManagement CorporateCon)	· 행정정보시스템의 운영과 관리를 책임지는 기관의 정보를 기술하는 요소로, 4개 세부요소로 구성됨 · 데이터세트 관리기준표의 '시스템 관리권한' 요소에 기재된 정보를 참조하여 작성함		
	세부요소	기관명 (CorporateName)	· 행정정보시스템 관리권한을 가진 기관명을 기술하는 요소
		기관코드 (CorporateCode)	· 행정정보시스템 관리권한을 가진 기관코드를 기술하는 요소
		부서명 (SectionName)	· 행정정보시스템 관리권한을 가진 부서명을 기술하는 요소
		부서코드 (SectionCode)	· 행정정보시스템 관리권한을 가진 부서코드를 기술하는 요소
접근권한 (InternalAccessScope)	· 데이터세트에 접근할 권한을 부여받은 주체를 기술하는 요소 · 데이터세트 관리기준표의 '접근권한' 요소에 기재된 정보를 참조하여 작성함		
비밀 (SecurityCon)	· 데이터세트의 보안과 관련된 비밀등급과 비밀기록물 책정근거 정보		
	세부요소	비밀분류 (SecurityLevel)	· 보안업무규정에 따른 기록물 비밀등급 분류를 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함 · 【1급】, 【2급】, 【3급】 중 택1
		비밀분류근거 (MandateofSecurity Level)	· 기록물(데이터세트)의 비밀등급을 설정한 근거를 기술하는 요소
		보호기간 (ProtectionPeriod)	· 보안업무규정에 따라 비밀로 보호되도록 지정된 기간을 기술하는 요소

공개 (ExternalAccessCon)	· 장기보존패키지 내 데이터세트의 이용 및 접근을 관리하고 통제하기 위한 권한 정보		
	세부요소	공개구분 (DisclosureType)	· 데이터세트의 공개 유형을 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함 · 【공개】, 【비공개】, 【부분공개】 중 택1 · 데이터세트 관리기준표의 '정보공개 구분' 요소에 기재된 정보를 참조하여 작성함
		비공개사유 (NonDisclosureReason)	· 기록물(데이터세트)을 공개하지 못하는 근거를 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함 · 다만, 이 요소는 '공개구분' 값이 【비공개】 혹은 【부분공개】 일 경우에만 작성하며, 비공개 사유(1호~8호) 중 해당하는 모든 항목을 나열하여 기재함 · 【1호】, 【2호】, 【3호】, 【4호】, 【5호】, 【6호】, 【7호】, 【8호】 중 택1
		공개제한부분 (LimitedContents)	· 비공개 대상 정보가 일부 포함되어 공개할 수 없는 데이터 범위를 기술하는 요소로, '공개구분' 요소 값이 【부분공개】 일 경우에만 작성함
		공개대상계층 (DisclosureLevel)	· 데이터세트의 공개 여부가 적용되는 계층적 수준을 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함 · 【단위기능】, 【테이블】, 【컬럼】 중 택1 · '공개구분' 요소 값이 【공개】 일 경우 【단위기능】을 선택함 · '공개구분' 요소 값이 【부분공개】 인 경우 【테이블】 또는 【컬럼】을 선택함 · '공개구분' 값이 【비공개】 일 경우 작성하지 않음
		공개대상범위 (DisclosureCoverage)	· 데이터세트의 공개 범위를 기술하는 요소 · '공개대상계층' 요소 값이 【테이블】, 【컬럼】 일 경우에만 작성함

- 생산기간(UsagePeriodCon)

· 데이터세트가 생산된 기간 정보를 기술하는 요소로, <표 35>와 같이 2개 최하위요소로 구성됨

· 생산기간은 실제 이관된 데이터세트에 포함된 데이터 중 가장 처음 생산된 시점(생산일시)부터 가장 마지막에 생산된 시점(종료일시)까지의 시간 범위를 뜻함(예: 생산일시: 2024-01-01, 종료일시: 2024-12-31). 여기서 종료일시는, 이 시점 이후로 데이터세트가 더 이상 변경되지 않는 고정 시점이 됨

· 생산기간을 구체적으로 알 수 없는 경우, 연도만 기재할 수 있음

· 데이터세트 종합관리시스템 이관기능에서 이관대상 데이터세트의 생산/종료연도 입력 가능함

<표 35> 생산기간(UsagePeriodCon) 하위요소의 구성

최하위요소	내용
생산일시 (UsageStartDateTime)	· 데이터세트의 생산이 시작된 시점을 기술하는 요소 · 데이터세트에 포함된 데이터 중 가장 처음 생산된 시점
종료일시 (UsageEndDateTime)	· 데이터세트의 생산이 종료된 시점을 기술하는 요소 · 데이터세트에 포함된 데이터 중 가장 마지막에 생산된 시점

- 보존기간정보(RetentionPeriodCon)

· 데이터세트의 보존기간 정보를 기술하는 요소로, <표 36>과 같이 3개 최하위요소로 구성됨

· 데이터세트가 시스템에 있는 동안에는 변경될 가능성이 있어 보존기간의 기산일 산정이 모호함. 이에 기록관으로 이관되는 시점을 기산일로 삼는 것을 제안함

<표 36> 보존기간정보(RetentionPeriodCon) 하위요소의 구성

최하위요소	내용
보존기간 (RetentionPeriod)	· 데이터세트의 법률적·행정적·역사적 가치에 따라 의무적으로 보존하여야 하는 기간을 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함 · 【1년】, 【3년】, 【5년】, 【10년】, 【30년】, 【준영구】, 【영구】 중 택1 · 데이터세트 관리기준표의 '보존기간' 요소에 기재된 정보를 참조하여 작성함
보존기간책정사유 (ReasionOfRetention Period)	· 데이터세트의 보존기간을 책정한 근거를 기술하는 요소 · 데이터세트 관리기준표의 '보존기간 책정사유' 요소에 기재된 정보를 참조하여 작성함
기산일 (RetentionStartDate)	· 데이터세트에 책정된 보존기간의 기산 시점을 기술하는 요소 · 데이터세트 관리기준표의 '기산일' 요소에 기재된 정보를 참조하여 작성함

- 처분(DispositionCon)

· 데이터세트의 보존기간 만료 후 수행되는 처분(보존기간 재책정, 공개재분류, 폐기 등) 정보를 기술하는 요소로, <표 37>과 같이 2개 최하위요소로 구성됨

<표 37> 처분(DispositionCon) 하위요소 구성

최하위요소	내용
처분제약사항 (DispositionConstraint)	· 데이터세트의 처분을 일시적으로 보류하거나 제한해야 하는 사유와 기간을 기술하는 요소 · 데이터세트 관리기준표의 '처분의 제약 발생사항' 요소에 기재된 정보를 참조하여 작성함
처분방법 (DispositionMethod)	· 데이터세트의 보존기간 및 처분 제약 기간이 경과한 후 적용되는 처분 방식을 기술하는 요소 · 데이터세트 관리기준표의 '처분방법' 요소에 기재된 정보를 참조하여 작성함

- 매체(Medium)

· 데이터세트가 저장 또는 보존된 매체 정보를 기술하는 요소

· 별도의 하위 항목을 가지지 않으며, 기록물을 외부 저장매체에 저장한 경우에만 작성함(논리적 SIP에 해당)

· 데이터세트에는 종종 첨부파일이 구성요소로 포함됨. 시스템의 아키텍처에 따라 첨부파일을 BLOB, CLOB 등의 데이터 타입으로 데이터베이스 테이블 내에 직접 저장하거나 첨부파일을 별도의 외부 저장매체에 저장하고 테이블에는 해당 파일의 위치를 가리키는 경로(File Path) 정보를 저장하는 방식이 있음. 후자의 경우, 파일의 저장 경로가 없으면 데이터와 첨부파일 간의 논리적 연결이 끊어지게 됨

· URN(UUID, DOI 등), URL(path(파일경로), ftp, http) 등을 작성함

- 관리이력(ManagementHistoryCon)

· 기록관리 전 과정에 걸쳐 기록물의 상태에 영향을 미치는 모든 관리행위에 대한 이력 정보를 기술하는 요소로, <표 38>과 같이 5개 최하위요소, 8개 세부요소로 구성됨

<표 38> 관리이력(UsagePeriodCon) 하위요소의 구성

최하위요소	내용		
관리유형 (EventType)	· 데이터세트의 상태에 영향을 미치는 관리행위 유형을 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함 · 【생산기관간 인계】, 【생산기관간 인수】, 【기록관으로 이관】, 【기록관에서 인수】, 【기록관간 이관】, 【기록관간 인수】, 【영구기록물관리기관으로 이관】, 【영구기록물관리기관에서 인수】, 【공개재분류】, 【보존기간 재책정】, 【보류】, 【폐기결정】, 【대체사본 보존결정】, 【폐기실행】, 【비밀재분류】, 【접근범위 재책정】, 【기록물관리기관에서 추가 등록】 중 택1		
관리설명 (EventDescription)	· 데이터세트의 상태에 영향을 미치는 관리행위에 대한 설명을 기술하는 요소		
관리일시 (EventDateTime)	· 데이터세트의 상태에 영향을 미치는 관리행위 발생 일시를 기술하는 요소		
관리행위자 (EventAgentCon)	· 데이터세트의 상태에 영향을 미치는 관리행위자 정보를 기술하는 요소		
	세부요소	기관명 (CorporateName)	· 데이터세트의 상태에 영향을 미치는 모든 관리행위에 책임이 있거나 관련이 있는 기관명을 기술하는 요소
		기관코드 (CorporateCode)	· 데이터세트의 상태에 영향을 미치는 모든 관리행위에 책임이 있거나 관련이 있는 기관코드를 기술하는 요소
		부서명 (SectionName)	· 데이터세트의 상태에 영향을 미치는 모든 관리행위에 책임이 있거나 관련이 있는 부서명을 기술하는 요소
		부서코드 (SectionCode)	· 데이터세트의 상태에 영향을 미치는 모든 관리행위에 책임이 있거나 관련이 있는 부서코드를 기술하는 요소
		개인명 (PersonName)	· 데이터세트의 상태에 영향을 미치는 모든 관리행위에 책임이 있거나 관련이 있는 개인명을 기술하는 요소
		직위(직급)명 (PositionTitle)	· 데이터세트의 상태에 영향을 미치는 모든 관리행위에 책임이 있거나 관련이 있는 담당자의 직위 및 직급을 기술하는 요소
변경요소 (ChangedElementCon)	· 데이터세트의 상태에 영향을 미치는 관리행위에 의해 메타데이터 값이 변할 경우 변경된 요소명과 그 이전값을 기술하는 요소		
	세부요소	변경요소명 (ChangedElement Name)	· 데이터세트의 상태에 영향을 미치는 관리행위가 발생했을 때 값이 변경된 메타데이터 요소명을 기술하는 요소 · 동일 명칭의 혼동을 방지하기 위해, 변경된 요소명은 최상위 요소부터 변경요소까지 전체 경로를 작성함 (예: 데이터세트정보-권한-데이터보유권한-기관명, 데이터세트정보-공개-공개구분)
		변경요소이전값 (ChangedElement Value)	· 데이터세트의 상태에 영향을 미치는 관리행위가 발생했을 때 값이 변경된 메타데이터 요소의 이전값을 기술하는 요소

- 보존이력(PreservationHistoryCon)

· 기록물관리기관으로 기록물이 인수된 이후 행해진 모든 보존처리행위에 대한 이력정보를 기술하는 요소로, <표 39>와 같이 4개 최하위요소, 6개 세부요소로 구성됨

<표 39> 보존이력(PreservationHistoryCon 하위요소 구성

최하위요소	내용		
보존처리유형 (PreservationAction Type)	· 데이터세트의 원형을 유지한 채 수명을 연장하기 위한 보존처리행위 유형을 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함 · 【매체수록(광디스크)】, 【복원】, 【보존포맷 변환】, 【장기보존패키지 변환】, 【바이러스 검출】, 【바이러스 치료】 중 택1		
보존처리설명 (PreservationAction Description)	데이터세트 원본의 상태에 대한 정보, 보존처리 사유 및 근거 등에 대한 설명을 기술하는 요소		
보존처리일시 (PreservationAction Date)	데이터세트에 보존처리행위가 취해진 일시를 기술하는 요소		
보존처리행위자 (PreservationAction AgentCon)	기록물(데이터세트)에 보존처리행위를 행한 행위자 정보		
	세부요소	기관명 (CorporateName)	데이터세트의 보존처리행위에 관여하거나 책임이 있는 기관명을 기술하는 요소
		기관코드 (CorporateCode)	데이터세트의 보존처리행위에 관여하거나 책임이 있는 기관코드를 기술하는 요소
		부서명 (SectionName)	데이터세트의 보존처리행위에 관여하거나 책임이 있는 부서명을 기술하는 요소
		부서코드 (SectionCode)	데이터세트의 보존처리행위에 관여하거나 책임이 있는 부서코드를 기술하는 요소
		개인명 (PersonName)	데이터세트의 보존처리행위에 관여하거나 책임이 있는 개인명을 기술하는 요소
		직위(직급)명 (PositionTitle)	데이터세트의 보존처리행위에 관여하거나 책임이 있는 담당자의 직위 및 직급을 기술하는 요소

- 컴포넌트(ComponentCon)

· 데이터세트 내 컴포넌트 정보를 기술하는 요소로, <표 40>과 같이 9개 최하위요소, 2개 세부요소로 구성됨

· 컴포넌트는 데이터세트에 포함된 첨부파일로, 데이터세트의 필수 구성요소이므로 함께 보존해야 함

· 데이터세트에 포함된 첨부파일 수에 따라 '컴포넌트' 하위요소를 반복 작성할 수 있음

<표 40> 컴포넌트(ComponentCon) 하위요소의 구성

최하위요소	내용		
컴포넌트유형 (ComponentType)	· 컴포넌트의 유형을 기술하는 요소로, 사전에 정의된 값 중 해당하는 값을 선택함 · 【원문】, 【보존포맷】, 【붙임파일】, 【재현정보】, 【기타】 중 택1 · 【기타】: 데이터세트 백업본(dump) 등을 말함 · SIP 내에서 컴포넌트 유형별로 폴더를 구성하여 관리해야 함 (예: 【보존포맷】 유형의 파일은 Preservation Format 폴더를 생성하여 저장함)		
컴포넌트제목 (ComponentTitle)	· 컴포넌트의 제목을 기술하는 요소		
컴포넌트개요 (ComponentOverview)	· 컴포넌트의 주요 내용과 성격 등에 대한 설명을 기술하는 요소		
컴포넌트포맷 (ComponentFormatCon)	· 컴포넌트의 파일포맷 정보를 기술하는 요소		
	세부요소	포맷명 (FormatName)	· 컴포넌트의 파일포맷명을 기술하는 요소
		포맷버전 (FormatVersion)	· 컴포넌트의 파일포맷 버전을 기술하는 요소
컴포넌트크기 (ComponentSize)	· 컴포넌트의 파일 크기를 기술하는 요소 · Byte 단위로 작성함		
컴포넌트위치 (ComponentLocation)	· 컴포넌트가 저장된 위치 정보를 기술하는 요소 · URN(UUID, DOI 등), URL(path(파일경로), ftp, http) 등		
컴포넌트세부식별자 (ComponentDetailedID)	· 단일 컴포넌트에 여러 데이터세트가 포함된 경우, 이를 구분하기 위한 세부 경로를 기술하는 요소 · '컴포넌트유형' 요소값이 【보존포맷】이고, 보존포맷이 ZIP과 같은 압축 구조일 경우 내부에 여러 데이터세트 파일이 존재할 수 있음. 단순히 ZIP 파일 단위만 기록하면 개별 데이터세트를 구분하기 어려우므로, 메타데이터 작성 시 ZIP 내부의 경로명(path)을 기재하도록 함 · 작성예시: zipfile.zip!/dataset1/file1.dat		
컴포넌트CI식별자 (ComponentCIID)	· 컴포넌트가 장기보존패키지 내에서 실제 어떤 파일과 대응되는지를 식별하기 위해 PackagingInformation.xml의 ManifestItem에 부여된 ManifestItemID 속성값을 기입하는 요소		
ManifestItemID참조 (ManifestIDRef)	· 컴포넌트가 장기보존패키지 내 어떤 기록물과 대응되는지 식별하기 위해 ContentInformation의 ContentItem에 부여된 ContentID 속성값을 기입하는 요소		

라. 장기보존패키지의 '기술정보(Descriptive Information)' 설계

○ 기술정보 설계 시 고려사항

- 기술정보(Descriptive Information, 이하 DI)는 OAIS 참조모형 정보패키지(Information Package)의 구성요소로써, 이용자가 패키지와 그 구성요소에 접근할 수 있도록 설계된 색인(index)과 같음

  · DI는 이용자가 정보를 검색(finding), 요청(ordering), 획득(retrieving)하는 전 과정을 지원하기 위해, 패키지와 그 구성요소에 대한 접근을 지원하는 정보로 구성되어야 함

- 패키지와 그 구성요소에 접근하려는 이용자의 행위는 '검색', '열람', '분석', '청구' 등 네 가지로 구분할 수 있음(<표 41> 참조). 각 행위는 DI가 담아야 할 핵심 정보 범주에 해당함

<표 41> DI 핵심 정보 범주

구분	내용
검색	· 이용자가 원하는 기록을 발견하기 위한 행위 · 검색을 위한 메타데이터는 이용자가 검색 질의로 활용할 수 있는 속성으로 구성되어야 함
열람	· 이용자가 검색을 통해 발견한 기록물의 내용을 검토하여 필요 여부를 판단하는 행위 · 열람을 위한 메타데이터는 이용자가 기록의 내용과 맥락을 파악하여, 검색 결과 중 찾는 기록물이 있는지 확인할 수 있도록 구성되어야 함
분석	· 이용자가 기록을 연구·업무·활용 목적에 맞게 가공하고 해석하는 행위 · 분석을 위한 메타데이터는 이용자가 기록의 구조적·기술적 특성을 파악하여 분석 가능성을 판단할 수 있도록 구성되어야 함 · 데이터 영역, 데이터 유형, 데이터 형식(파일 포맷), 데이터 구조, 데이터 구축연도, 파일 크기, 관련 소프트웨어 및 도구 등
청구	· 이용자가 원하는 기록을 특정한 뒤, 이를 공식적으로 요청하는 행위 · 청구를 위한 메타데이터는 이용자가 기록을 정확히 식별하고, 접근 권한과 공개여부 등을 사전에 파악할 수 있도록 구성되어야 함 · 고유식별자(콘텐츠 ID 등), 접근 권한 및 공개 구분, 담당 기관, 요청 절차 정보 등

○ 장기보존 메타데이터 요소 검토

- DI는 패키지 내부의 정보를 활용하여 이용자가 외부에서 해당 패키지를 검색하고 접근할 수 있도록 구성한 메타데이터임

  · 새로운 요소를 별도로 설계하지 않고, 패키지에 포함된 기존 메타데이터 중 접근성 확보에 유용한 항목을 선별하여 활용함(일반적으로 DI는 패키지의 장기보존 메타데이터에서 파생됨)

- 이에 앞서 제시한 검색, 열람, 분석, 청구 등 네 가지 범주를 기준으로, 장기보존 메타데이터 요소 중 관련 요소를 선별하여 DI로 관리하고자 함(<표 42> 참조)

  · 매핑 수준은 ●(확실하게 매핑되는 경우) 2점, ◎(매핑될 가능성이 있는 경우) 1.5점, ○(간접적으로 매핑되는 경우) 1점으로 산정함

  · 하나의 요소는 검색, 열람, 분석, 청구 등 복수의 기준에 중복 매핑될 수 있으며, 이 경우 해당 기준별 점수를 합산하여 평가함

  · 총점이 3점을 초과하는 요소는 필수적으로 기록·관리되어야 하는 핵심요소로 선정함

<표 42> DI 핵심 정보 범주 및 장기보존메타데이터(안) 요소 간 대응관계

차상위요소명	하위요소명	최하위요소명	세부요소명	검색	열람	분석	청구
객체메타데이터 (ObjectCon)	객체유형 (ObjectType)				●	◎	○
	객체버전 (ObjectVersion)				○	◎	
	객체유형설명 (ObjectType Description)				○		
	객체생성일시 (ObjectCreation Date)			◎	●	◎	
	패키징범위 (PackagingScope)				○		
생산자 (CreatorCon)	생산자유형 (CreatorType)			◎	●	○	●
	기관명 (CorporateName)			●	●	○	●
	기관코드 (CorporateCode)			○			○
	부서명 (SectionName)						
	부서코드 (SectionCode)						
	개인명 (PersonName)						
기록관리식별자 (Record ManagementID)				●			●
기록물명 (RecordTitle)				●	●		●

시스템정보 (SystemInfoCon)	시스템명 (SystemName)			●	●		○
	시스템식별자 (SystemID)			●			●
	시스템개요 (SystemOverview)			●	●		
	시스템구축연도 (SystemImplementationYear)			◎	●	◎	
	시스템폐기연도 (SystemDecommissioningYear)			◎	●	◎	
	DBMS정보 (DBMSInfoCon)	DBMS명/버전 (DBMSNameVersion)			○	●	
		업무DB명 (FunctionDBName)				◎	
	연계시스템 (InterfacedSystemCon)	연계유형 (InterfaceType)				○	
		연계시스템명 (InterfacedSystemName)		○	◎	○	
		연계시스템식별자 (InterfacedSytemID)					
		연계시스템 보유기관명 (InterfacedSystemCoporateName)					
		연계시스템 보유기관코드 (InterfacedSystemCoporateID)					
		연계형태 (InterfaceType)	연계방식 (InterfaceMethod)			●	
			연계방향 (InterfaceDirection)				
	산출물 (DeliverableCon)	산출물유형 (DeliverableType)		○	●	◎	
		산출물제목 (DeliverableTitle)		●	●		●
		산출물포맷 (Deliverable FormatCon)	포맷명 (FormatName)		○	●	○
			포맷버전 (FormatVersion)			◎	
		산출물크기 (DeliverableSize)				●	
		산출물위치 (Deliverable Location)			○		
		ManifestID참조 (ManifestIDRef)					●
		산출물CI식별자 (DeliverableCIID)					●

데이터세트정보 (DatasetInfoCon)	단위기능정보 (UnitFunctionCon)	단위기능식별자 (UnitFunctionID)		●			●
		단위기능명 (UnitFunctionName)		●	●	○	●
		단위기능개요 (UnitFunction Overview)		●	●		
		주제어 (SubjectWords)		●	●		
	단위기능범위 (UnitFunction ScopeCon)	DB질의문 (RetrievalQuery)				●	
		적용범위 (ApplicationScope)			○	●	○
		적용범위관련정보 (Application ScopeInfo)					
	권한 (RightsCon)	데이터보유권한 (DataOwing CorporateCon)	기관명 (CorporateName)	●	◎		●
			기관코드 (CorporateCode)	○			○
			부서명 (SectionName)				
			부서코드 (SectionCode)				
		시스템관리권한 (System Management CorporateCon)	기관명 (CorporateName)	●	◎		●
			기관코드 (CorporateCode)	○			○
			부서명 (SectionName)				
			부서코드 (SectionCode)				
		접근권한 (InternalAccess Scope)		◎	●		●
		비밀 (SecurityCon)	비밀분류 (SecurityLevel)	◎	●		●
			비밀분류근거 (Mandateof SecurityLevel)		●		
			보호기간 (ProtectionPeriod)		●		
		공개 (ExternalAccess Con)	공개구분 (DisclosureType)	◎	●		●
			비공개사유 (NonDisclosure Reason)		●	○	
			공개제한부분 (LimitedContents)				
			공개대상계층 (DisclosureLevel)			◎	
			공개대상범위 (Disclosure Coverage)				

데이터세트정보 (DatasetInfoCon)	생산기간 (UsagePeriodCon)	생산일시 (UsageStartDate Time)		◎	●	◎	
		종료일시 (UsageEndDate Time)		◎	●	◎	
	보존기간정보 (RetentionPeriodCon)	보존기간 (RetentionPeriod)			●	○	●
		보존기간책정사유 (ReasionOf RetentionPeriod)			●		
		기산일 (RetentionStart Date)			○	○	
	처분 (DispositionCon)	처분제약사항 (Disposition Constraint)			●	○	
		처분방법 (DispositionMethod)					
	매체 (Medium)				○	●	
	관리이력 (Management HistoryCon)	관리유형 (EventType)			●		
		관리설명 (EventDescription)			●		
		관리일시 (EventDateTime)			●	○	
		관리행위자 (EventAgentCon)	기관명 (CorporateName)		●		○
			기관코드 (CorporateCode)		○		◎
			부서명 (SectionName)				
			부서코드 (SectionCode)				
			개인명 (PersonName)				
			직위(직급)명 (PositionTitle)				
		변경요소 (Changed ElementCon)	변경요소명 (Changed ElementName)		◎		
			변경이전값 (Changed ElementValue)		◎		

데이터세트정보 (DatasetInfoCon)	보존이력 (Preservation HistoryCon)	보존처리유형 (Preservation ActionType)			●		
		보존처리설명 (Preservation ActionDescription)			●		
		보존처리일시 (Preservation ActionDate)			●		
		보존행위자 (Preservation ActionAgentCon)	기관명 (CorporateName)		●		●
			기관코드 (CorporateCode)		○		◎
			부서명 (SectionName)				
			부서코드 (SectionCode)				
			개인명 (PersonName)				
			직위(직급)명 (PositionTitle)				
	컴포넌트 (ComponentCon)	컴포넌트유형 (ComponentType)		○	●	◎	
		컴포넌트제목 (ComponentTitle)		●	●		●
		컴포넌트개요 (Component Overview)			●		
		컴포넌트포맷 (Component FormatCon)	포맷명 (FormatName)		○	●	○
			포맷버전 (FormatVersion)			◎	
		컴포넌트크기 (ComponentSize)				●	
		컴포넌트위치 (Component Location)			○		
		컴포넌트세부위치 (Component DetailLocation)			○		
		컴포넌트CI식별자 (ComponentCIID)					●
		ManifestItemID 참조 (ManifestIDRef)					●

- 점수 매핑 결과를 토대로 총점 3점 이상을 획득한 요소를 핵심요소로 선별함. 각 핵심요소의 기능, 적용 근거, 관리 필요성 등을 종합적으로 제시함으로써 핵심 요소의 선정 타당성을 확인하고, 향후 기록관리 설계의 근거를 명확히 함

○ 객체메타데이터(ObjectCon) 차상위요소 중 기술정보(DI) 활용 요소

- 객체유형, 객체버전, 객체유형설명, 객체생성일시, 패키징범위로 구성됨. 그중 객체유형, 객체생성일시 하위요소만이 기준점수를 통과함

<표 43> 객체메타데이터(ObjectCon) 차상위요소 중 기술정보(DI) 활용 가능 요소

하위요소명	내용	
객체유형 (ObjectType)	열람	이용자가 장기보존패키지의 유형을 파악하여 필요한 기록 여부를 판단함
	분석	장기보존패키지 유형에 따라 적합한 분석 방식을 결정하는 데 활용될 수 있음
	청구	장기보존패키지 청구 시 대상 기록을 특정하기 위한 구분값으로 활용될 수 있음
객체생성일시 (ObjectCreationDate)	검색	특정 시점에 생성된 장기보존패키지(SIP)를 찾기 위한 주요 질의 조건으로 활용 가능함
	열람	장기보존패키지(SIP)의 생성 시점을 통해 생성 맥락과 시점 파악, 이용자가 맥락을 이해하는 데 도움을 줌
	분석	생성 시점 정보를 통해 기록의 구조적, 기술적 특성 분석에 필요한 시간적 맥락을 제공할 수 있음

○ 생산자(CreatorCon) 차상위요소 중 기술정보(DI) 활용 요소

- 생산자유형, 기관명, 기관코드, 부서명, 부서코드, 개인명으로 구성됨. 그중 생산자유형, 기관명 하위요소만이 기준점수를 통과함

<표 44> 생산자(CreatorCon) 차상위요소 중 기술정보(DI) 활용 가능 요소

하위요소명	내용	
생산자유형 (CreatorType)	검색	이용자가 생산주체 유형을 검색도구로 활용할 수 있음
	열람	생산 주체의 성격(보유기관·관리기관·기타)을 통해 이용자가 기록의 생성 배경과 맥락을 이해하는 데 도움을 줌
	분석	생산자 유형에 따라 데이터의 구조, 관리 방식 등이 달라지므로 분석 시 출처별 특성 비교에 활용가능함
	청구	청구 시 담당 기관 또는 책임 주체를 특정하는 근거로 활용되어, 접근 경로 및 요청 절차 판단에 적합함
기관명 (CorporateName)	검색	이용자가 기관명을 검색도구로 활용할 수 있음
	열람	기관명을 통해 데이터세트의 생산 주체와 배경을 파악할 수 있어 기록의 맥락 이해에 적합함
	분석	생산 현황, 관리체계 등 생산기관이 분석대상으로 활용될 수 있음
	청구	기관명은 요청 대상 기관을 명확히 특정할 수 있는 근거로 작용하여 청구 절차 수행에 필수적임

❍ 기록관리식별자(RecordManagementID) 차상위요소 중 기술정보(DI) 활용 요소

- 검색: 고유 식별코드를 통해 특정 패키지를 직접 검색·조회할 수 있어 주요 검색 조건으로 활용됨
- 청구: 청구 시 대상 패키지를 명확히 특정하기 위한 식별정보로 활용됨

❍ 기록물명(RecordTitle) 차상위요소 중 기술정보(DI) 활용 요소

- 열람: 이용자가 기록물명을 검색도구로 활용하여 특정한 장기보존패키지를 찾는 데 활용됨
- 분석: 기록물명을 통해 기록의 내용 및 범위에 대한 파악이 가능함
- 청구: 청구 요청 대상 기록을 특정하기 위한 기본정보로 활용됨

❍ 시스템정보(SystemInfoCon) 차상위요소중 기술정보(DI) 활용 요소

- 시스템명, 시스템식별자, 시스템개요, 시스템구축연도, 시스템폐기연도, DBMS 정보, 연계시스템, 산출물로 구성됨. 그중 아래 하위요소만이 기준점수를 통과함

<표 45> 시스템정보(SystemInfoCon) 차상위요소 중 기술정보(DI) 활용 가능 요소

하위요소명	내용	
시스템명 (SystemName)	검색	이용자가 행정정보시스템명을 검색도구로 활용하여 관련 데이터세트를 찾는 데 활용함
	열람	시스템명을 통해 행정정보시스템의 내용 및 범위에 대한 파악이 가능함
	청구	청구 대상이 되는 행정정보시스템을 식별하기 위한 정보로 활용될 수 있음
시스템식별자 (SystemID)	검색	행정정보시스템의 고유 코드로 검색도구로 활용하여 정확한 시스템 기반 검색 수행에 활용됨
	청구	청구 대상 시스템을 명확히 특정하고 요청 범위를 한정하기 위한 식별정보로 활용됨
시스템개요 (SystemOverview)	검색	시스템개요 내 주요 기능·목적 등의 키워드가 검색 질의에 노출될 수 있어 검색 조건으로 활용됨
	열람	시스템개요를 통해 행정정보시스템의 기능, 역할 등을 파악할 수 있어 내용 이해에 적합함
시스템구축연도 (System ImplementationYear)	검색	특정 연도에 구축된 시스템을 기준으로 검색할 수 있음
	열람	구축 시점을 통해 시스템의 생성 배경과 운영 시작 시기를 파악할 수 있어 열람 기준에 부합함
	분석	구축 연도를 통해 기술 환경, 개발 시기의 행정 환경 등 분석에 필요한 시점 정보를 파악할 수 있음
시스템폐기연도 (System DecommissioningYear)	검색	특정 시점에 폐기된 시스템을 기준으로 검색할 수 있음
	열람	폐기 시점을 통해 시스템의 운영 종료 시기와 관리 이력을 파악할 수 있어 열람 기준에 부합함
	분석	폐기 연도를 통해 시스템 수명주기, 데이터 이전 시점 등 분석에 활용 가능한 시간 정보를 파악할 수 있음

- DBMS정보(DBMSInfoCon) 하위요소

<표 46> DBMS정보(DBMSInfoCon) 하위요소 중 기술정보(DI) 활용 가능 요소

최하위요소명	내용	
DBMS명/버전 (DBMSName Version)	열람	사용 중인 DBMS의 종류와 버전 정보를 통해 시스템의 기술 환경과 데이터 관리 체계를 이해할 수 있음
	분석	DBMS의 종류과 버전 정보를 통해 데이터 구조, 포맷 호환성, 분석 도구 적용 가능성 등을 판단할 수 있어 기술적 분석에 활용됨

- 연계시스템(Interfaced SystemCon) 하위요소

<표 47> 연계시스템(InterfacedSystemCon) 하위요소 중 기술정보(DI) 활용 가능 요소

최하위요소명	내용	
연계시스템명 (Interfaced SystemName)	검색	연계시스템명을 검색도구로 활용하여 관련 행정정보시스템을 파악할 수 있음
	열람	연계시스템명을 통해 행정정보시스템과 연계된 타 시스템명을 파악할 수 있음
	분석	연계시스템명을 통해 시스템 간 연계구조, 상호작용 관계 등을 분석할 수 있음

- 산출물(DeliverableCon) 하위요소

<표 48> 산출물(DeliverableCon) 하위요소 중 기술정보(DI) 활용 가능 요소

최하위요소명		내용	
산출물유형 (DeliverableType)		검색	이용자가 특정 산출물 형태(제안요청서, ERD, 사용자매뉴얼 등)를 기준으로 관련 기록을 검색할 수 있음
		열람	산출물 유형을 통해 기록의 성격, 활용목적 등에 대한 파악이 가능함
		분석	산출물 유형별로 기술 수준, 문서 구조 등을 비교·분석할 수 있음
산출물제목 (DeliverableTitle)		검색	이용자가 산출물의 제목 검색도구로 활용하여 원하는 자료를 직접 검색할 수 있음
		열람	산출물 제목을 통해 산출물의 내용과 범위를 직관적으로 파악할 수 있어 내용 이해에 적합함
		청구	청구대상 산출물을 명확히 특정하기 위한 기본 정보로 활용됨
산출물포맷 (Deliverable FormatCon)	포맷명 (Format Name)	열람	산출물 포맷명을 통해 산출물의 열람 가능 여부와 접근 방식을 확인할 수 있음
		분석	데이터 구조와 처리 방식 분석에 필요한 기술적 특성을 제공함
		청구	포맷 정보를 통해 제공가능한 형태와 변환 여부를 판단할 수 있어 청구절차 수행에 활용가능함

❍ 데이터세트정보(DatasetInfoCon) 차상위요소중 기술정보(DI) 활용 요소

- 단위기능정보, 단위기능범위, 권한, 생산기간, 보존기간정보, 처분, 매체, 관리이력, 보존이력, 컴포넌트로 구성됨. 그중 아래 하위요소만이 기준점수를 통과함

- 단위기능정보(UnitFunctionCon) 하위요소

<표 49> 단위기능정보(UnitFunctionCon) 하위요소 중 기술정보(DI) 활용 가능 요소

최하위요소명	내용	
단위기능식별자 (UnitFunctionID)	검색	단위기능의 고유 식별코드를 검색도구로 활용함
	청구	청구 시 대상 단위기능을 명확히 특정하기 위한 식별정보로 활용됨
단위기능명 (UnitFunctionName)	검색	이용자가 단위기능명을 검색도구로 활용하여 관련 데이터세트를 검색함
	열람	단위기능명을 통해 데이터세트가 수행하는 업무 목적과 내용을 파악함
	분석	단위기능별로 데이터 구조, 업무 관계를 비교·분석할 수 있음
	청구	단위기능명을 통해 요청 대상 데이터세트를 특정하기 위한 식별정보로 활용됨
단위기능개요 (UnitFunctionOverview)	검색	단위기능개요 내 주요 업무, 기능 등의 키워드가 검색 질의에 노출될 수 있어 검색 조건으로 활용됨
	열람	단위기능개요를 통해 기능의 목적, 수행 범위, 주요 업무 등을 파악함
주제어 (SubjectWords)	검색	이용자가 입력하는 핵심 검색어로 직접 활용되어 관련 데이터세트를 효과적으로 탐색할 수 있음
	열람	주제어를 통해 단위기능의 주요 내용과 범주 등을 파악함

- 단위기능범위(UnitFunctionScopeCon) 하위요소

<표 50> 단위기능범위(UnitFunctionScopeCon) 하위요소 중 기술정보(DI) 활용 가능 요소

최하위요소명	내용	
적용범위 (ApplicationScope)	열람	적용범위를 통해 단위기능이 다루는 데이터 영역과 포함 테이블을 확인할 수 있음
	분석	적용범위 정보는 데이터 구조와 기능 간 연계 관계를 분석하는 데 활용됨
	청구	적용범위를 통해 청구 대상 데이터의 범위를 한정할 수 있음

- 권한(RightsCon) 하위요소

<표 51> 권한(RightsCon) 하위요소 중 기술정보(DI) 활용 가능 요소

최하위요소명		내용	
데이터보유 권한 (DataOwing CorporateCon)	기관명 (Corporate Name)	검색	이용자가 기관명을 기준으로 관련 데이터세트를 검색함
		열람	기관명을 통해 데이터의 생산, 관리주체를 확인할 수 있음
		청구	데이터보유권한을 지닌 기관명을 통해 청구 대상 기관을 특정함
시스템관리 권한 (System Management CorporateCon)	기관명 (Corporate Name)	검색	이용자가 시스템 관리기관명을 기준으로 관련 행정정보시스템을 검색함
		열람	기관명을 통해 시스템의 운영 주체와 관리 책임을 확인할 수 있음
		청구	시스템관리권한을 지닌 기관명을 통해 청구 대상 기관을 특정함
접근권한 (InternalAccessScope)		검색	접근권한 수준에 따라 이용가능한 데이터세트를 구분·검색할 수 있음
		열람	접근권한 정보를 통해 열람 가능 여부와 제한 조건을 확인함
		청구	접근권한은 청구 가능 범위와 절차를 판단하는 정보로 활용됨
비밀 (SecurityCon)	비밀분류 (Security Level)	검색	비밀등급을 기준으로 공개·비공개 기록을 구분하여 검색할 수 있음
		열람	비밀등급을 통해 열람 가능 여부와 접근 제한 수준을 확인할 수 있음
		청구	비밀분류는 청구 시 공개 가능성 및 절차를 판단하는 근거로 활용됨
공개 (SecurityCon)	공개구분 (Disclosure Type)	검색	공개 유형(공개·비공개·부분공개)을 기준으로 검색함
		열람	공개구분을 통해이용 가능한 데이터세트의 열람 가능 범위를 확인할 수 있음
		청구	공개 여부에 따라 청구 가능성을 판단함
	비공개 사유 (Non Disclosure Reason)	열람	비공개 결정의 근거를 통해 이용자가 열람 제한 사유를 확인함
		분석	비공개 사유 유형을 분석함으로써 기록물의 접근 제한 기준을 평가할 수 있음

- 생산기간(UsagePeriodCon) 하위요소

<표 52> 생산기간(UsagePeriodCon) 하위요소 중 기술정보(DI) 활용 가능 요소

최하위요소명	내용	
생산일시 (UsageStartDateTime)	검색	특정 시점에 생산된 데이터를 기준으로 검색할 수 있음
	열람	데이터의 최초 생성 시점을 통해 기록의 시작 맥락을 파악함
	분석	생산 시점을 통해 데이터의 구축시기 및 생성 배경을 분석할 수 있음
종료일시 (UsageEndDateTime)	검색	특정 기간 내 종료된 데이터를 검색 기준으로 활용할 수 있음
	열람	데이터의 종료 시점을 통해 기록의 완료 시기와 범위를 파악함
	분석	종료 시점을 통해 데이터 갱신 주기, 누적 기간 등을 분석할 수 있음

- 보존기간정보(RetentionPeriodCon) 하위요소

<표 53> 보존기간정보(RetentionPeriodCon) 하위요소 중 기술정보(DI) 활용 가능 요소

최하위요소명	내용	
보존기간 (RetentionPeriod)	열람	보존기간을 통해 기록의 관리 상태와 활용 가능 시점을 확인함
	분석	보존기간 정보를 통해 데이터의 가치, 법적·행정적 중요도 등을 파악할 수 있음
	청구	보존기간은 청구 가능 여부와 절차 판단의 기준이 되며, 청구 대상 기록의 존속 여부를 확인하는 데 활용됨

- 처분(DispositionCon) 하위요소

<표 54> 처분(DispositionCon) 하위요소 중 기술정보(DI) 활용 가능 요소

최하위요소명	내용	
처분제약사항 (DispositionConstraint)	열람	처분제약사항을 통해 데이터세트의 이용 제한 사유와 기간을 확인함
	분석	제약 사유 및 기간 정보를 통해 기록물의 관리 정책, 보존 결정 과정 등을 분석할 수 있음

- 매체(Medium) 하위요소

· 열람: 데이터세트가 저장된 매체 정보를 통해 접근가능성 및 보존 형태를 확인할 수 있음

· 분석: 매체 유형에 따라 저장 구조, 접근 방식, 보존 안정성 등을 비교·분석할 수 있음

- 관리이력(ManagementHistoryCon) 하위요소

<표 55> 관리이력(ManagementHistoryCon) 하위요소 중 기술정보(DI) 활용 가능 요소

최하위요소명		내용	
관리일시 (EventDateTime)		열람	관리일시를 통해 데이터세트의 영향을 가하는 요소를 확인함
		분석	관리행위의 시점을 기반으로 유지보수 주기 등 관리행위를 분석할 수 있음
관리행위자 (EventAgentCon)	기관명 (Corporate Name)	열람	관리행위자 기관명을 통해 데이터세트의 관리주체와 책임 기관을 확인함
		청구	관리기관 정보를 통해 청구 대상 또는 협조 기관을 특정할 수 있음

- 보존이력(PreservationHistoryCon) 하위요소

<표 56> 보존이력(PreservationHistoryCon) 하위요소 중 기술정보(DI) 활용 가능 요소

최하위요소명		내용	
보존행위자 (Preservation ActionAgentCon)	기관명 (Corporate Name)	열람	보존처리기관명을 통해 데이터세트의 보존행위 주체와 책임 기관을 확인함
		청구	보존처리기관명을 통해 청구 대상 기관을 특정할 수 있음

- 컴포넌트(ComponentCon) 하위요소

<표 57> 컴포넌트(ComponentCon) 하위요소 중 기술정보(DI) 활용 가능 요소

최하위요소명		내용	
컴포넌트유형 (ComponentType)		검색	이용자가 컴포넌트 유형(원문, 보존포맷 등)을 기준으로 관련 기록을 검색할 수 있음
		열람	컴포넌트 유형을 통해 데이터세트 내 파일의 성격을 파악할 수 있음
		분석	컴포넌트 유형별로 데이터 구성, 포맷 구조, 보존 방식 등을 구분·비교하여 기술적 특성을 분석할 수 있음
컴포넌트제목 (ComponentTitle)		검색	파일명을 기준으로 특정한 컴포넌트를 검색함
		열람	파일명을 통해 컴포넌트의 내용과 성격을 확인함
		청구	파일명은 청구 대상 컴포넌트를 특정하기 위한 기본정보로 활용됨
컴포넌트포맷 (Component FormatCon)	포맷명 (Format Name)	열람	파일의 형식 정보를 통해 해당 컴포넌트의 열람 가능 여부와 접근 방식을 확인할 수 있음
		분석	포맷명은 데이터 구조, 인코딩 방식, 기술적 속성 분석에 활용됨
		청구	포맷 정보를 통해 제공가능한 파일 형태와 변환 필요 여부를 판단하는 데 활용될 수 있음

❍ Dublin Core 및 EAD(Encoded Archival Description) 메타데이터 요소 매핑

- 국제적으로 널리 활용되는 메타데이터 표준인 Dublin Core와 EAD(Encoded Archival Description, 이하 EAD)를 매핑 기준으로 선정함

· 국제 메타데이터 표준과의 매핑을 통해 요소 간 의미의 일관성과 상호운용성을 확보함

- E-ARK(European Archival Records and Knowledge Preservation System)에서 정의한 정보패키지 모델 역시 Dublin Core와 EAD를 모두 포함함. 이는 두 표준이 국제적으로 디지털 자원 기술과 기록 기술의 핵심 참조 체계로 공인되어 있음을 보여줌

- Dublin Core는 웹상의 다양한 디지털 자원을 효율적으로 검색·관리하기 위해 개발된 표준 메타데이터임

· Dublin Core는 자원의 식별과 기술을 위해 정의된 15개 핵심 요소로 구성됨(<표 58> 참조)

<표 58> Dublin Core 요소

Dubin Core 요소명	내용
Title (제목)	자원의 명칭
Creator (제작자)	자원을 만든 개인, 단체
Subject (주제)	자원의 주제
Description (설명)	자원에 대한 요약, 설명
Publisher (발행자)	자원을 현재 형태로 이용가능하게 만든 주체 (배포자, 발행자)
Contributor (기여자)	Creator 이외에 자원제작에 기여한 개인, 단체
Date (날짜)	자원의 생애주기에 있는 사건과 관련된 특정 시점이나 기간
Type (유형)	자원의 성격(범주, 기능, 장르 유형)
Format (형식)	물리적/디지털적 양식(포맷)
Identifier (식별자)	고유하게 자원을 식별하는 정보(식별자-URL, DOIS, ISBN)
Source (출처)	기술된 자원이 파생된 관련 자원
Language (언어)	사용된 언어
Relation (관계)	관련된 자원
Coverage (범위)	자원의 공간적 적용범위 또는 자원이 적용되는 관할 구역
Rights (권한)	자원에 대한 권리 정보

· 해당 표준은 문헌정보학 분야에서 문헌의 식별·탐색·관리를 위해 서지 메타데이터로 활용되며, 자원의 내용과 속성을 기술하여 이용자가 자원을 이해하고 접근할 수 있도록 지원함

· 이는 DI의 목적 및 기능과 일치하며, 국제적으로 통용되는 요소 의미 체계를 제공할 수 있음

· 이에 DI는 Dublin Core를 매핑 기준으로 적용하여, DI 자체 요소가 국제 표준과 의미적으로 대응될 수 있도록 설계함

· 특히 DI 핵심 정보 범주 및 장기보존 메타데이터(안) 요소 간의 대응 관계에서 총점 3점 이상을 획득한 핵심요소를 중심으로 Dublin Core 요소와의 매핑을 수행함으로써, 실제 장기보존패키지 설계에 필요한 핵심 메타데이터만을 선별적으로 반영함(<표 59> 참조)

· Dublin Core에서 자원은 식별 또는 기술의 대상을 의미하므로, 본 내용에서는 자원이 기록물 개요정보에 해당하는 장기보존패키지와 데이터세트에 해당하는 단위기능 정보를 뜻함

<표 59> Dublin Core 및 장기보존 메타데이터(안) 요소 간 대응 관계

Dublin Core 요소명	장기보존 메타데이터(안) 기록물 개요정보	장기보존 메타데이터(안) 데이터세트 단위기능정보
Title(제목)	기록물명	데이터세트정보-단위기능정보-단위기능명
Creator (제작자)	생산자-기관명의 합집합	데이터세트정보-권한-데이터보유권한-기관명의 합집합
Subject (주제)	시스템정보-시스템명;데이터세트정보-단위기능정보-단위기능명의 합집합	데이터세트정보-단위기능정보-주제어
Description (설명)	시스템정보-시스템개요	데이터세트정보-단위기능정보-단위기능개요
Publisher (발행자)	데이터세트정보-보존이력-보존행위자-기관명	데이터세트정보-보존이력-보존행위자-기관명
Contributor (기여자)	· 생산자-생산자유형(데이터보유기관)-기관명의 합집합 · 생산자-생산자유형(시스템관리기관)-기관명의 합집합	· 데이터세트정보-권한-데이터보유권한-기관명의 합집합 · 데이터세트정보-권한-시스템관리권한-기관명의 합집합
Date (날짜)	객체메타데이터-객체생성일시	· 데이터세트정보-생산기간-생산일시 · 데이터세트정보-생산기간-종료일시
Type (유형)	'행정정보 데이터세트'로 고정값 설정	'행정정보 데이터세트 단위기능'으로 고정값 설정
Format (형식)	· 시스템정보-DBMS정보-DBMS명/버전의 합집합 · 시스템정보-산출물-산출물포맷-포맷명의 합집합	· 데이터세트정보-컴포넌트-컴포넌트포맷-포맷명의 합집합
Identifier (식별자)	기록관리식별자	데이터세트정보-단위기능정보-단위기능식별자
Source (출처)	· 시스템정보-시스템명 · 시스템정보-시스템식별자	시스템정보-시스템명
Language (언어)	'ko'로 고정값 설정	'ko'로 고정값 설정
Relation (관계)	시스템정보-연계시스템-연계시스템명의 합집합	· 데이터세트정보-컴포넌트-컴포넌트제목의 합집합 · 데이터세트정보-컴포넌트-컴포넌트제목의 합집합
Coverage (범위)	· 시스템정보-시스템구축연도 · 시스템정보-시스템폐기연도	· 데이터세트정보-단위기능범위-적용범위 · 데이터세트정보-보존기간정보-보존기간
Rights (권한)	· 생산자-생산자유형(데이터보유기관)-기관명의 합집합 · 생산자-생산자유형(시스템관리기관)-기관명의합집합	· 데이터세트정보-권한-접근권한 · 데이터세트정보-권한-비밀-비밀분류 · 데이터세트정보-권한-공개-공개구분

- Dublin Core와 장기보존 메타데이터(안) 간 요소 대응은, 장기보존패키지, 데이터세트의 구조적 특성을 고려하여 설정함

- Dublin Core의 각 요소는 선택적으로 사용될 수 있으며 하나의 요소가 여러 번 기술될 수 있음. 본 설계에서는 장기보존 메타데이터(안)에 따라 필요한 요소만을 우선 매핑하였으나, 향후 보존정책의 변화나 기술 환경의 확장에 따라 생략된 요소를 추가적으로 반영할 수 있음

<표 60> Dublin Core - Title(제목) 및 장기보존 메타데이터(안) 요소 간 대응 관계

구분	내용	
기록물 개요정보	정의	장기보존패키지를 대표하는 식별적 명칭
	대상	기록물명
	적용 근거	Dublin Core의 title은 자원에 부여된 명칭을 기술하는 요소임. 본 설계에서 자원은 장기보존패키지를 의미하므로, 장기보존패키지에 부여된 제목을 dc:title 값으로 매핑하는 것이 타당함
	예시	[기록물명] 인사정보기록
단위기능정보	정의	데이터세트를 대표하는 고유 명칭
	대상	데이터세트정보-단위기능정보-단위기능명
	적용 근거	Dublin Core의 title은 부여된 명칭을 기술하는 요소임. 본 설계에서 자원은 데이터세트정보를 의미하므로, 데이터세트를 식별하기 위해 부여되는 단위기능명을 dc:title 값으로 매핑하는 것이 타당함
	예시	[기록물명] 인사정보기록

<표 61> Dublin Core - Creator(제작자) 및 장기보존 메타데이터(안) 요소 간 대응 관계

구분	내용	
기록물 개요정보	정의	장기보존패키지를 생산한 주체
	대상	생산자-기관명의 합집합 (생산자유형이 '데이터보유기관'인 경우에만 해당)
	적용 근거	Dublin Core의 creator는 자원을 만든 개인이나 단체를 기술하는 요소임. 본 설계에서 자원은 장기보존패키지를 의미하므로, 장기보존패키지에 포함된 데이터세트의 생산자가 속한 기관명을 dc:creator 값으로 매핑하는 것이 타당함 또한 기관명의 합집합으로 구성되므로 각각의 기관명을 세미콜론(;)으로 구분함
	예시	[생산자-기관명]A기관;B기관;C기관
단위기능 정보	정의	데이터세트를 생산한 주체
	대상	권한-데이터보유권한-기관명의 합집합
	적용 근거	Dublin Core의 creator는 자원을 만든 개인이나 단체를 기술하는 요소임. 본 설계에서 자원은 데이터세트를 의미하므로, 데이터세트에 대한 보유권한을 가진 기관이 데이터를 생산한 것으로 추정, 데이터보유권한-기관명을 dc:creator 값으로 매핑하는 것이 타당함 또한 기관명의 합집합으로 구성되므로 각각의 기관명을 세미콜론(;)으로 구분함
	예시	[데이터세트정보-권한-데이터보유권한-기관명]A기관;B기관;C기관

<표 62> Dublin Core - Subject(주제) 및 장기보존 메타데이터(안) 요소 간 대응 관계

구분	내용	
기록물 개요정보	정의	장기보존패키지의 주제
	대상	시스템정보-시스템명;데이터세트정보-단위기능정보-단위기능명의 합집합
	적용 근거	· Dublin Core의 subject는 자원의 주제를 기술하는 요소임. 본 설계에서 자원은 장기보존패키지를 의미하므로, 장기보존패키지 자체의 주제어를 기술해야 함. 그러나 장기보존패키지의 주제어를 입력하는 메타데이터 요소가 별도로 정의되어 있지 않음. 따라서 행정정보시스템명과 데이터세트 단위기능명의 합집합이 장기보존패키지를 대표하는 주제어로서 기능하는 것으로 판단, dc:subject 값으로 매핑하는 것이 타당함 · 또한 시스템명 및 단위기능명의 합집합으로 구성되므로 각각의 명칭을 세미콜론(;)으로 구분함
	예시	[시스템정보-시스템명]A시스템;[데이터세트-단위기능정보-단위기능명]B기능;C기능;D기능;E기능;F기능
단위기능정보	정의	데이터세트의 주제
	대상	데이터세트정보-단위기능정보-주제어
	적용 근거	· Dublin Core의 subject는 자원의 주제를 기술하는 요소임. 본 설계에서 자원은 데이터세트정보이며, 단위기능정보의 주제어는 해당 데이터세트가 다루는 개념을 표현함. 이는 subject 요소의 목적과 정확히 일치하므로, 단위기능주제어를 dc:subject에 매핑하는 것이 타당함 · 단, 장소명, 기간처럼 공간·시간 범위를 표현하는 값은 subject가 아니라 coverage 값으로 매핑되는 것이 효율적임
	예시	[데이터세트정보-단위기능정보-주제어]AAA;BBB;CCC;DDD

<표 63> Dublin Core - Subject(주제) 및 장기보존 메타데이터(안) 요소 간 대응 관계

구분	내용	
기록물 개요정보	정의	장기보존패키지의 범위, 내용, 구성요소 등에 대한 요약 및 설명
	대상	시스템정보-시스템개요
	적용 근거	· Dubin Core의 description은 자원에 대한 요약이나 내용 설명 정보를 기술하는 요소임. 본 설계에서 자원은 장기보존패키지를 의미하므로, 해당 자원을 포괄적으로 설명할 수 있는 시스템개요를 description 요소에 매핑하는 것이 타당함 · 또한 description 요소는 여러 개를 값으로 가질 수 있기 때문에, 필요에 따라 시스템 개요 이외의 다양한 보조 설명을 추가적으로 기술할 수 있음. 따라서 기본적으로는 시스템개요를 대표 description 값으로 제시하되, 기타 설명 정보도 수용할 수 있도록 함
	예시	[시스템정보-시스템개요] A시스템은 …. 이다.
단위기능정보	정의	데이터세트의 내용, 용도 등에 대한 요약 및 설명
	대상	데이터세트정보-단위기능정보-단위기능개요
	적용 근거	· Dubin Core의 description은 자원에 대한 요약이나 내용 설명 정보를 기술하는 요소임. 본 설계에서 자원은 데이터세트정보를 의미하므로, 단위기능의 업무활용목적과 같이 해당 자원을 포괄적으로 설명할 수 있는 단위기능개요를 대표적인 description 요소에 매핑하는 것이 타당함 · 또한 description 요소는 여러 개를 값으로 가질 수 있기 때문에, 필요에 따라 다양한 보조 설명을 추가적으로 기술할 수 있음. 따라서 기본적으로는 단위기능개요를 대표 description 값으로 제시하되, 기타 설명 정보도 수용할 수 있도록 함
	예시	[데이터세트정보-단위기능정보-단위기능개요] A기능은 …. 이다.

<표 64> Dublin Core - Publisher(발행자) 및 장기보존 메타데이터(안) 요소 간 대응 관계

구분	내용	
기록물 개요정보	정의	장기보존패키지를 외부에 공개 및 배포할 수 있는 주체
	대상	데이터세트정보-보존이력-보존행위자-기관명 (모든 데이터세트의 해당 요소 중에서 가장 최근에 보존처리유형이 '장기보존패키지 변환'인 경우에만 해당)
	적용 근거	Dublin Core의 publisher는 자원을 현재 형태로 이용 가능하게 만든 주체를 기술하는 요소임. 본 설계에서 자원은 장기보존패키지를 의미하므로, 데이터세트의 보존처리행위에 관여하거나 책임이 있는 기관, 즉 현재의 장기보존패키지를 최종적으로 이관받아 외부에 공개 및 배포할 수 있는 기관명을 dc:publisher 값으로 매핑하는 것이 타당함.
	예시	[데이터세트정보-보존이력-보존행위자-기관명]A기관
단위기능정보	정의	데이터세트를 공식적으로 서비스 및 배포하는 주체
	대상	데이터세트정보-보존이력-보존행위자-기관명 (모든 데이터세트의 해당 요소 중에서 가장 최근에 보존처리유형이 '장기보존패키지 변환'인 경우에만 해당)
	적용 근거	Dublin Core의 publisher는 자원을 현재 형태로 이용 가능하게 만든 주체를 기술하는 요소임. 본 설계에서 자원은 데이터세트를 의미하므로, 데이터세트의 보존처리행위에 관여하거나 책임이 있는 기관, 즉 현재 데이터세트를 공식적으로 서비스하고 배포할 권한을 가진 기관명을 dc:publisher 값으로 매핑하는 것이 타당함.
	예시	[데이터세트정보-보존이력-보존행위자-기관명]A기관

<표 65> Dublin Core - Contributor(기여자) 및 장기보존 메타데이터(안) 요소 간 대응 관계

구분	내용	
기록물 개요정보	정의	Creator 이외에 장기보존패키지 생성과정에 기여한 기관, 부서
	대상	· 생산자-생산자유형(데이터보유기관)-기관명의 합집합 · 생산자-생산자유형(시스템관리기관)-기관명의 합집합
	적용 근거	· Dublin Core의 contributor는 creator 이외에 자원의 제작에 기여한 개인이나 단체를 기술하는 요소임. 본 설계에서 자원은 장기보존패키지를 의미하므로, 장기보존패키지에 속한 데이터세트의 생산에 직접적으로 연관된 보유기관과 해당 시스템을 관리하는 기관을 dc:contributor 값으로 매핑하는 것이 타당함 · 또한 기관명의 합집합으로 구성되므로 각각의 기관명을 세미콜론(;)으로 구분함
	예시	· <dc:contributor>[생산자-생산자유형-기관명(데이터보유기관)]A기관;B기관;C기관</dc:contributor> · <dc:contributor>[생산자-생산자유형-기관명(시스템관리기관)]D기관;E기관;F기관</dc:contributor>
단위기능 정보	정의	Creator 이외에 데이터세트 생성 및 이관 과정에 기여한 기관, 부서
	대상	· 데이터세트정보-권한-데이터보유권한-기관명의 합집합 · 데이터세트정보-권한-시스템관리권한-기관명의 합집합
	적용 근거	· Dublin Core의 contributor는 creator 이외에 자원의 제작에 기여한 개인이나 단체를 기술하는 요소임. 본 설계에서 자원은 데이터세트를 의미하므로, 데이터세트에 대한 보유권한을 가진 기관과 행정정보시스템의 관리권한을 가진 기관을 dc:contributor 값으로 매핑하는 것이 타당함 · 또한 기관명의 합집합으로 구성되므로 각각의 기관명을 세미콜론(;)으로 구분함
	예시	· <dc:contributor>[데이터세트정보-권한-데이터보유권한-기관명]A기관;B기관;C기관</dc:contributor> · <dc:contributor>[데이터세트정보-권한-시스템관리권한-기관명]D기관;E기관;F기관</dc:contributor>

<표 66> Dublin Core - Date(날짜) 및 장기보존 메타데이터(안) 요소 간 대응 관계

구분	내용	
기록물 개요정보	정의	장기보존패키지의 생애주기에 있는 사건과 관련된 특정 시점이나 기간
	대상	객체메타데이터-객체생성일시
	적용 근거	· Dublin Core의 date는 자원의 생애주기에 연관된 사건의 시점 또는 기간을 기술하는 요소임. 본 설계에서 자원은 장기보존패키지를 의미하므로, 장기보존패키지의 최초 구축 시점을 대표 이벤트로 선택하여, dc:date에 매핑하는 것이 타당함. 이는 시스템이 언제 성립·가동을 시작했는지를 일관되게 식별하기 위한 대표 시점 값임 · Dubin Core는 ISO 8601 방식을 사용하므로, “YYYY-MM-DDThh:mm;ssZ”의 기입방식을 준수하도록 함
	예시	[객체메타데이터-객체생성일시] 2025-10-03T13:45:00+09:00
단위기능정보	정의	데이터세트의 생애주기에 있는 사건과 관련된 특정 시점이나 기간
	대상	· 데이터세트정보-생산기간-생산일시 · 데이터세트정보-생산기간-종료일시
	적용 근거	· Dublin Core의 date는 자원의 생애주기에 연관된 사건의 시점이나 기간을 기술하도록 정의됨. 본 설계에서 자원은 데이터세트를 의미하므로, 데이터세트의 생산일시 및 종료일시를 dc:date에 각각 매핑하는 것이 타당함. 만약 dcterms:temporal 속성을 활용할 경우, 기간(생산~폐기)을 표현할 수 있으므로, “생산일시/종료일시”와 같이 기술할 수 있음 · Dubin Core는 ISO 8601 방식을 사용하므로, “YYYY-MM-DDThh:mm;ssZ”의 기입방식을 준수하도록 함
	예시	· <dc:date>[데이터세트정보-생산기간-생산일시]2025-10-03T13:45:00+09:00 </dc:date> · <dc:date>[데이터세트정보-생산기간-종료일시]2026-10-02T13:45:00+09:00 </dc:date>

<표 67> Dublin Core - Type(유형) 및 장기보존 메타데이터(안) 요소 간 대응 관계

구분	내용	
기록물 개요정보	정의	장기보존패키지의 성격(범주, 기능, 장르, 유형 등)
	대상	행정정보 데이터세트
	적용 근거	Dublin Core의 type은 자원의 성격, 범주, 기능, 혹은 장르 유형을 기술하는 요소임. 본 설계에서 자원은 장기보존패키지를 의미하므로, 해당 장기보존패키지는 “행정정보 데이터세트”라는 유형을 지니고 있음. 따라서 이를 dc:type 값으로 매핑하는 것이 타당함
	예시	<dc:type> 행정정보 데이터세트 </dc:type>
단위기능정보	정의	데이터세트의 성격(범주, 기능, 장르, 유형 등)
	대상	행정정보 데이터세트 단위기능
	적용 근거	Dublin Core의 type은 자원의 성격, 범주, 기능, 혹은 장르 유형을 기술하는 요소임. 본 설계에서 자원은 데이터세트를 의미하므로, “행정정보 데이터세트 단위기능”이라는 유형을 dc:type 값으로 매핑하는 것이 타당함
	예시	<dc:type> 행정정보 데이터세트 단위기능</dc:type>

<표 68> Dublin Core - Format(형식) 및 장기보존 메타데이터(안) 요소 간 대응 관계

구분	내용	
기록물 개요정보	정의	장기보존패키지의 물리적/디지털적 형식(포맷)
	대상	시스템정보-DBMS정보-DBMS명/버전의 합집합 시스템정보-산출물-산출물포맷-포맷명의 합집합
	적용 근거	· Dublin Core의 format은 자원의 물리적 혹은 디지털적 형태(포맷)를 기술하는 요소임. 본 설계에서 자원은 장기보존패키지를 의미하므로, 패키지를 구성하는 데이터의 저장 및 접근 환경을 규정하는 DBMS 종류와 버전, 그리고 최종 산출물의 파일포맷명을 기술하는 것이 적절함. 따라서 행정정보시스템에서 사용 중인 DBMS명/버전과 산출물의 포맷명을 dc:format 값으로 매핑하는 것이 타당함 · 또한 DBMS명버전 또는 포맷명의 합집합으로 구성되므로 각각의 명칭을 세미콜론(;)으로 구분함
	예시	· <dc:format>[시스템정보-DBMS정보-DBMS명/버전]Oracle19c Edition;MySQL 0.0</dc:format> · <dc:format>[시스템정보-산출물-산출물포맷-포맷명] hwp;pdf;xlsx;dax</dc:format>
단위기능정보	정의	데이터세트의 물리적/디지털적 형식(포맷)
	대상	데이터세트정보-컴포넌트-컴포넌트포맷-포맷명의 합집합
	적용 근거	· Dublin Core의 format은 자원의 물리적 혹은 디지털적 형태(포맷)를 기술하는 요소임. 본 설계에서 자원은 데이터세트를 의미하므로, 데이터세트를 구성하는 각 컴포넌트의 포맷명이 자원의 디지털 형식을 나타냄. 따라서 컴포넌트의 파일포맷명을 dc:format 값으로 매핑하는 것이 타당함 · 또한 포맷명의 합집합으로 구성되므로 각각의 포맷명을 세미콜론(;)으로 구분함
	예시	<dc:format>[데이터세트정보-컴포넌트-컴포넌트포맷-포맷명] hwp;pdf;xlsx;dax</dc:format>

<표 69> Dublin Core - Identifier(식별자) 및 장기보존 메타데이터(안) 요소 간 대응 관계

구분	내용	
기록물 개요정보	정의	장기보존패키지를 식별하는 정보
	대상	기록관리식별자
	적용 근거	Dublin Core의 identifier은 주어진 맥락에서 자원을 고유하게 식별하는 정보를 기술하는 요소임. 본 설계에서 자원은 장기보존패키지를 의미하므로, 장기보존패키지 식별코드를 의미하는 기록관리식별자를 dc:identifier에 매핑하는 것이 타당함
	예시	[기록관리식별자]DAT 1030000 003354B0000 2025 00000 00001
단위기능정보	정의	데이터세트를 식별하는 정보
	대상	데이터세트정보-단위기능정보-단위기능식별자
	적용 근거	Dublin Core의 identifier는 주어진 맥락에서 자원을 고유하게 식별하는 정보를 기술하는 요소임. 본 설계에서 자원은 데이터세트를 의미하므로, 데이터세트를 식별할 수 있는 단위기능식별자를 dc:identifier에 매핑하는 것이 타당함
	예시	[데이터세트정보-단위기능정보-단위기능식별자]NAK_CAMS_01

<표 70> Dublin Core - Source(출처) 및 장기보존 메타데이터(안) 요소 간 대응 관계

구분	내용	
기록물 개요정보	정의	장기보존패키지가 파생된 관련 자원
	대상	· 시스템정보-시스템명 · 시스템정보-시스템식별자
	적용 근거	Dublin Core의 source는 기술된 자원이 파생된 관련 자원, 즉 출처를 기술하는 요소임. 본 설계에서 자원은 장기보존패키지를 의미하므로, 패키지가 생성된 근원 자원은 해당 패키지에 포함된 데이터세트를 산출한 행정정보시스템으로 해석됨. 따라서 행정정보시스템의 공식명과 시스템식별자를 dc:source 값으로 매핑하는 것이 타당함
	예시	· <dc:source>[시스템정보-시스템명]A시스템</dc:source> · <dc:source>[시스템정보-시스템식별자]003354B0000</dc:source>
단위기능정보	정의	데이터세트가 파생된 관련 자원
	대상	시스템정보-시스템명
	적용 근거	Dublin Core의 source는 기술된 자원이 파생된 관련 자원, 즉 출처를 기술하는 요소임. 본 설계에서 자원은 데이터세트를 의미하므로, 데이터세트의 생성 근원은 이를 산출한 행정정보시스템으로 해석됨. 따라서 행정정보시스템의 공식명을 dc:source 값으로 매핑하는 것이 타당함
	예시	<dc:source>[시스템정보-시스템명]A시스템</dc:source

<표 71> Dublin Core - Language(언어) 및 장기보존 메타데이터(안) 요소 간 대응 관계

구분	내용	
기록물 개요정보	정의	장기보존패키지에 사용된 언어
	대상	장기보존패키지 전체
	적용 근거	Dublin Core의 language는 자원에 사용된 언어를 기술하는 요소임. 본 설계에서 자원은 장기보존패키지를 의미하나, 장기보존 메타데이터에는 별도로 언어 정보를 기술하는 요소가 존재하지 않음. 따라서 장기보존패키지의 기본 기술언어를 한국어로 간주하여, ISO 639 Alpha-2 코드 체계를 따라 'ko' 값을 dc:language로 고정 입력하는 것이 타당함
	예시	<dc:language>ko</dc:language>
단위기능정보	정의	데이터세트에 사용된 언어
	대상	데이터세트 전체
	적용 근거	Dublin Core의 language는 자원에 사용된 언어를 기술하는 요소임. 본 설계에서 자원은 데이터세트를 의미하나, 장기보존 메타데이터에는 언어 정보를 별도로 기술하는 요소가 존재하지 않음. 따라서 데이터세트의 기본 기술언어를 한국어로 간주하고, ISO 639 Alpha-2 표준에 따라 'ko' 값을 dc:language로 고정 입력하는 것이 타당함
	예시	<dc:language>ko</dc:language>

<표 72> Dublin Core - Relation(관계) 및 장기보존 메타데이터(안) 요소 간 대응 관계

구분	내용	
기록물 개요정보	정의	장기보존패키지와 관련된 다른 자료
	대상	시스템정보-연계시스템-연계시스템명의 합집합
	적용 근거	· Dublin Core의 relation은 기술 대상 자원과 다른 자원 간의 관계를 기술하는 요소임. 본 설계에서 자원은 장기보존패키지를 의미하므로, 패키지와 연계되어 있는 행정정보시스템 간의 관계를 표현하기 위해 연계시스템명을 dc:relation 값으로 매핑하는 것이 타당함 · 관련 자원은 URI로 식별하는 것이 권장되나, 연계시스템의 경우 일반적으로 URL 형태로 제공되지 않으며, 이용자가 검색·열람 등의 업무를 수행할 때 시스템명을 중심으로 식별하는 특성을 가짐. 따라서 공식 식별체계에 부합하는 문자열 형태로 연계시스템명을 기술하는 것이 합리적임 · 만약 해당 연계시스템이 존재하지 않는 경우에는 dc:relation 요소를 생략할 수 있음 · 또한 연계시스템명의 합집합으로 구성되므로 각각의 시스템명을 세미콜론(;)으로 구분함
	예시	[시스템정보-연계시스템-연계시스템명]A시스템;B시스템;C시스템
단위기능 정보	정의	데이터세트와 관련된 다른 자료
	대상	· 데이터세트정보-컴포넌트-컴포넌트제목의 합집합 (컴포넌트유형이 '붙임파일'인 경우에만 해당, 최대5개) · 데이터세트정보-컴포넌트-컴포넌트제목의 합집합 (컴포넌트유형이 '재현정보'인 경우에만 해당, 최대5개)
	적용 근거	· Dublin Core의 relation은 기술 대상 자원과 다른 자원 간의 관계를 기술하는 요소임. 본 설계에서 자원은 데이터세트를 의미하므로, 데이터세트와 연관된 붙임파일, 재현정보 등을 나타내는 컴포넌트의 파일명을 dc:relation 값으로 매핑하는 것이 타당함 · 만약 해당 컴포넌트 파일이 존재하지 않는 경우에는 dc:relation 요소를 생략할 수 있음 · 또한 컴포넌트제목의 합집합으로 구성되므로 각각의 명칭을 세미콜론(;)으로 구분함
	예시	· <dc:coverage>[데이터세트정보-컴포넌트-컴포넌트명(붙임)a.hwp;b.pdf;c.docx;d.xslx</dc:coverage> · <dc:coverage>[데이터세트정보-컴포넌트-컴포넌트명(재현)]e.xslt;f.xml;g.css;h.html</dc:coverage>

<표 73> Dublin Core - Coverage(범주) 및 장기보존 메타데이터(안) 요소 간 대응 관계

구분	내용	
기록물 개요정보	정의	장기보존패키지의 공간적/시간적 적용 범위
	대상	· 시스템정보-시스템구축연도 · 시스템정보-시스템폐기연도
	적용 근거	· Dublin Core의 coverage는 자원의 공간적/시간적 주제, 적용범위, 관할 구역 등을 기술하는 요소임. 본 설계에서 자원은 장기보존패키지를 의미하므로, 패키지의 생성 및 관리가 연계된 행정정보시스템의 운영 기간을 시간적 적용범위로 해석함. 따라서 행정정보시스템이 구축되어 운영을 시작한 연도와 공식적으로 폐기된 연도를 dc:coverage 값으로 매핑하는 것이 타당함
	예시	· <dc:coverage>[시스템정보-시스템구축연도]2025</dc:coverage> · <dc:coverage>[시스템정보-시스템폐기연도]2050</dc:coverage>
단위기능정보	정의	데이터세트의 공간적/시간적 적용 범위
	대상	· 데이터세트정보-단위기능범위-적용범위 · 데이터세트정보-보존기간정보-보존기간
	적용 근거	· Dublin Core의 coverage는 자원의 공간적/시간적 주제, 적용범위, 관할 구역 등을 기술하는 요소임. 본 설계에서 자원은 데이터세트를 의미하므로, 데이터세트가 적용되는 데이터베이스 내 테이블 또는 데이터 영역의 범위를 공간적 적용범위로, 데이터세트의 의무적 보존기간을 시간적 적용범위로 해석함. 따라서 적용범위와 보존기간을 dc:coverage 값으로 매핑하는 것이 타당함 · 또한 다수의 적용범위를 나타낼 경우, 각각의 명칭을 세미콜론(;)으로 구분함
	예시	· <dc:coverage>[데이터세트정보-단위기능범위-적용범위]A테이블;B테이블;C테이블D열;D테이블(2024-2025)</dc:coverage> · <dc:coverage>[데이터세트정보-보존기간정보-보존기간]영구</dc:coverage>

<표 74> Dublin Core - Rights(권한) 및 장기보존 메타데이터(안) 요소 간 대응 관계

구분	내용	
기록물 개요정보	정의	장기보존패키지에 대한 권리 정보
	대상	· 생산자-생산자유형(데이터보유기관)-기관명의 합집합 · 생산자-생산자유형(시스템관리기관)-기관명의 합집합
	적용 근거	· Dublin Core의 rights는 자원에 대한 권리 정보를 기술하는 요소임. 본 설계에서 자원은 장기보존패키지를 의미하므로, 해당 패키지의 이용 및 관리 권한을 보유한 생산자 기관 정보를 권리 주체로 매핑함. 즉, 데이터세트의 생산자(데이터보유기관, 시스템관리기관)-기관명을 dc:rights 값으로 매핑하는 것이 타당함 · 또한 기관명의 합집합으로 구성되므로 각각의 기관명을 세미콜론(;)으로 구분함
	예시	· <dc:rights>[생산자-생산자유형-기관명(데이터보유기관)]A기관;B기관;C기관</dc:rights> · <dc:rights>[생산자-생산자유형-기관명(시스템관리기관)]D기관;E기관;F기관</dc:rights>
단위기능정보	정의	데이터세트에 대한 권리 정보
	대상	· 데이터세트정보-권한-접근권한 · 데이터세트정보-권한-비밀-비밀분류 · 데이터세트정보-권한-공개-공개구분
	적용 근거	· Dublin Core의 rights는 자원에 대한 권리 정보를 기술하는 요소임. 본 설계에서 자원은 데이터세트를 의미하므로, 해당 데이터세트의 이용·공개와 관련된 권한 정보를 명시해야 함. 따라서 데이터세트의 접근성에 관한 요소인 접근권한, 비밀분류, 공개구분을 dc:rights 값으로 매핑하는 것이 타당함
	예시	· <dc:rights>[데이터세트정보-권한-접근권한]부서담당자;기록관담당자;시스템관리자</dc:rights> · <dc:rights>[데이터세트정보-권한-비밀-비밀분류]1급</dc:rights> · <dc:rights>[데이터세트정보-권한-공개-공개구분]비공개</dc:rights>

- EAD는 기록물의 계층적 구조와 맥락 정보를 표현하기 위해 개발된 XML 기반 메타데이터 표준임
  · 기록물 검색도구를 위한 메타데이터 표준으로 활용되고 있으며, 기록관리기관 간 정보 공유와 통합 검색을 지원하고 웹 기반 접근성을 향상시키는 데 기여함
  · EAD는 기록 컬렉션 전체와 개별 컴포넌트를 기술할 수 있는 계층적 데이터 구조를 제공하여, DI가 패키지 내부의 기록 간 관계와 구조적 맥락을 체계적으로 표현할 수 있도록 지원할 수 있음
  · EAD는 Dublin Core의 15개 요소에 대응하는 요소를 반영함과 동시에, 대응하지 않는 일부 핵심 요소를 보완하는 기능으로 사용되었음(<표 75> 참조). 해당 요소의 활용 여부는 추가 논의가 필요함
  · 따라서 EAD는 구조적 맥락을 보완하는 선택적 기술요소로 서비스될 수 있음
- DI 핵심 정보 범주 및 장기보존 메타데이터(안) 요소 간의 대응 관계에서 총점 3점 이상을 획득한 핵심요소를 중심으로 수행한 EAD 요소 매핑은 다음과 같음

<표 75> EAD 및 장기보존 메타데이터(안) 요소 대응관계

구분	장기보존 메타데이터(안) 요소	EAD 요소명 (및 속성)
기록물 개요정보	객체메타데이터-객체유형	<did> <genreform>
	객체메타데이터-객체생성일시	<did><unitdate>
	생산자-생산자유형	<origination> - localtype 속성
	생산자-기관명	<origination> - localtype 속성 <corpname><part>
	기록관리식별자	<did><unitid>
	기록물명	<did><unittitle>
	시스템정보-시스템명	<repository> - localtype 속성 <corpname>
	시스템정보-시스템식별자	<repository> - identifier 속성 <corpname>
	시스템정보-시스템개요	<scopecontent><p>
	시스템정보-시스템구축연도	<did><unitdatestructured><daterange><fromdate>
	시스템정보-시스템폐기연도	<did><unitdatestructured><daterange><todate>
	시스템정보-DBMS정보-DBMS명/버전	<phystech> <p>
	시스템정보-연계시스템-연계시스템명	<relatedmaterial><corpname>
	시스템정보-산출물-산출물유형	<odd><p>
	시스템정보-산출물-산출물제목	<odd><unittitle>
	시스템정보-산출물-산출물포맷-포맷명	<odd><phystech><list><item>
단위기능정보	데이터세트정보-단위기능정보-단위기능식별자	<did><unitid>
	데이터세트정보-단위기능정보-단위기능명	<did><unittitle>
	데이터세트정보-단위기능정보-단위기능개요	<scopecontent><p>
	데이터세트정보-단위기능정보-주제어	<controlaccess><subject>
	데이터세트정보-단위기능범위-적용범위	<scopecontent><p>
	데이터세트정보-권한-데이터보유권한-기관명	<repository><corpname><part>
	데이터세트정보-권한-시스템관리권한-기관명	<custodhist><corpname><part>
	데이터세트정보-권한-접근권한	<accessrestrict><p>
	데이터세트정보-권한-비밀-비밀분류	<accessrestrict><p>
	데이터세트정보-권한-공개-공개구분	<accessrestrict><p>
	데이터세트정보-권한-공개-비공개사유	<accessrestrict><p>
	데이터세트정보-생산기간-생산일시	<did> <unitdatestructured> <daterange> <fromdate>
	데이터세트정보-생산기간-종료일시	<did><unitdatestructured> <daterange><todate>
	데이터세트정보-보존기간정보-보존기간	<appraisal><p>
	데이터세트정보-처분-처분제약사항	<appraisal><p>
	데이터세트정보-매체	<phystech><p>
	데이터세트정보-관리이력-관리일시	<custodhist> - standarddate 속성 <unitdatestructured><datesingle>
	데이터세트정보-관리이력-관리행위자-기관명	<custodhist><corpname><part>
	데이터세트정보-보존이력-보존행위자-기관명	<processinfo><corpname><part>
	데이터세트정보-컴포넌트-컴포넌트유형	<dao>
	데이터세트정보-컴포넌트-컴포넌트제목	<dao><daodesc><unittitle>
	데이터세트정보-컴포넌트-컴포넌트포맷-포맷명	<dao><daodesc><phystech><p>

○ 기술정보(DI) 설계(안)

- 앞서 제시한 Dublin Core 및 EAD와 장기보존 메타데이터(안) 요소 매핑 결과를 종합하여, 최종 기술정보 메타데이터 요소를 제시함(<표 76> 참조)

<표 76> 최종 기술정보 메타데이터 요소

차상위 요소명	하위 요소명	최하위 요소명	세부 요소명	Dublin Core 요소	EAD 요소(및 속성)
객체 메타데이터	객체유형			-	<did><genreform>
	객체생성 일시			Date	<did><unitdate>
생산자	생산자유형			-	<origination> - localtype 속성
	기관명			Creator, Contributor, Rights	<origination> - localtype 속성 <corpname><part>
기록관리 식별자				Identifier	<did><unitid>
기록물명				Title	<did><unittitle>
시스템 정보	시스템명			Subject, Source	<repository> - localtype 속성 <corpname>
	시스템 식별자			Source	<repository> - identifier 속성 <corpname>
	시스템개요			Description	<scopecontent><p>
	시스템 구축연도			Coverage	<did><unitdatestructured> <daterange><fromdate>
	시스템 폐기연도			Coverage	<did><unitdatestructured> <daterange><todate>
	DBMS 정보	DBMS명/버전		Type	<phystech><p>
	연계시스템	연계시스템명		Relation	<relatedmaterial><corpname>
	산출물	산출물유형		-	<odd><p>
		산출물제목		-	<odd><unittitle>
		산출물포맷	포맷명	Type	<odd><phystech><list><item>

데이터 세트 정보	단위기능 정보	단위기능 식별자		Identifier	<did><unitid>
		단위기능명		Title	<did><unittitle>
		단위기능개요		Description	<scopecontent><p>
		주제어		Subject	<controlaccess><subject>
	단위기능 범위	적용범위		Coverage	<scopecontent><p>
	권한	데이터보유권한	기관명	Creator, Subject, Contributor	<repository><corpname><part>
		시스템관리권한	기관명	Contributor	<custodhist><corpname><part>
		접근권한		Rights	<accessrestrict><p>
		비밀	비밀분류	Rights	<accessrestrict><p>
		공개	공개구분	Rights	<accessrestrict><p>
			비공개 사유	-	<accessrestrict><p>
	생산기간	생산일시		Date	<did><unitdatestructured> <daterange><fromdate>
		종료일시		Date	<did><unitdatestructured> <daterange><todate>
	보존기간 정보	보존기간		Coverage	<appraisal><p>
	처분	처분제약사항		-	<appraisal><p>
	매체			-	<phystech><p>
	관리이력	관리일시		-	<custodhist> - standarddate 속성 <unitdatestructured><datesingle>
		관리행위자	기관명	-	<custodhist><corpname><part>
	보존이력	보존행위자	기관명	Publisher	<processinfo><corpname><part>
	컴포넌트	컴포넌트유형		-	<dao>
		컴포넌트제목		Relation	<dao><daodesc><unittitle>
		컴포넌트포맷	포맷명	Type	<dao><daodesc><phystech><p>

- 해당 최종 기술정보 메타데이터 요소를 바탕으로 장기보존패키지 XML 및 XSD를 설계함

· 각 xml 요소에 값을 기입할 때, 장기보존 메타데이타와 연계된 요소가 존재한다면, [차상위요소부터 전체 요소명]을 기입하고 실제 값을 기입함

마. 장기보존패키지(NEO3) 생성 및 검증 방안 설계

○ 개요: 장기보존패키지(NEO3)의 생성 단계는 다음과 같은 5단계로 이루어짐

- **(1단계) 데이터세트 및 메타데이터 설정**: 대상 범위·스냅샷 정책 결정, 식별자/버전, 해시 알고리즘, PDI(Preservation Description Information)/CI(Content Information)/DI(Descriptive Information) 요소 매핑 등을 설정함
- **(2단계) 전자기록물(원문·보존포맷) 정합화**: 원문(예: DB dump)과 보존포맷(데이터세트/xsd+xml, 재현, 붙임) 구성 및 연계함
- **(3단계) 기록관리 메타데이터 확정**: Metadata.xml에 원문·보존포맷 파일 목록과 해시를 수록하여 참조 일관성을 확보함
- **(4단계) 행정전자서명 적용**: Metadata.xml을 입력값으로 서명값을 생성(Signature.xml)하고, 인증서(Certificate.cer)를 첨부함
- **(5단계) 패키징 수행**: 폴더 구조 정렬 및 파일 배치, 패키지 인벤토리·해시값 산출 후 PackagingInformation.xml에 기록함

○ (1단계) 데이터세트 및 메타데이터 설정

<그림 49> 장기보존패키지(NEO3) 생성 1단계

① Metadata.xml의 PreservationDescInfo는 '행정정보 데이터세트 종합관리시스템' 에서 해당 시스템의 관리기준표로부터 정보를 수집함. 또는 모든 정보를 채우지 못하는 경우 자체 관리하는 행정정보시스템, 행정정보시스템 산출물, 정보자원관리시스템으로부터 수집하여 작성함

② DescriptiveMetadata요소는 PreservationDescInfo요소로부터 추출 또는 요약하여 설정함

③ DB로부터 텍스트 형식의 dump 파일을 추출함

④ 데이터세트와 재현 관련 정보는 데이터베이스로부터 추출함

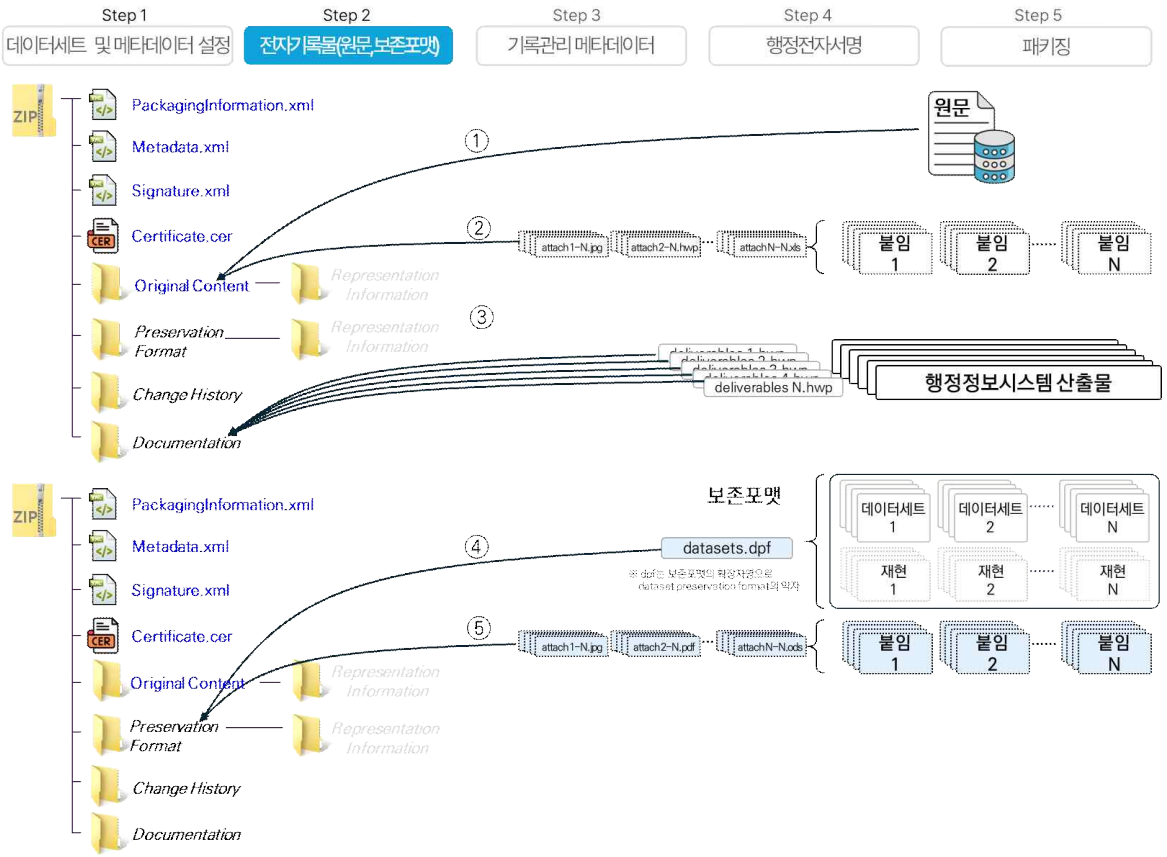
⑤ 붙임 파일 즉 데이터베이스에서 관리되지 않고, 데이터베이스의 특정 필드값과 디스크 스토리지의 파일들이 연결되어 있는 경우, 붙임 파일들이 관리되는 방식이 다르므로 기관별로 알맞은 방식을 채택해야 한다. 본 연구에서는 저장되는 유형을 구분하여 방안을 제시하고자 함

⑥ 행정정보시스템의 산출물은 각 기관의 행정정보시스템 관리자 또는 정보자원관리시스템으로 받을 수 있음

○ (2단계) 전자기록물(원문·보존포맷) 정합화

- DB에는 실제 데이터가 저장되지 않고, URL, URN 등의 식별자로만 표시되어 있고 별도로 저장되어 있는 파일들을 붙임 파일이라 함. 업무수행 시 첨부자료의 형태로 이러한 붙임 파일들이 대량/대용량으로 저장·관리되고 있는 실정이며, 현재 데이터세트의 기록물의 범위에 이 붙임 파일을 구성요소로 정의하고 있으므로, 장기보존패키지에서 수용할 수 있어야 함

- 데이터세트의 경우에는 정보시스템 구축 후에 행정정보시스템 산출물들이 생산됨. 제안요청서, 용역설계서, 시스템구성도, ERD, 테이블 명세서, 사용자/운영자 매뉴얼, 메뉴구조도, 화면설계서, 연계시스템 정의서 등이 대표적인 정보시스템 산출물임. 데이터세트를 이해하는 크게 도움이 되는 정보이므로 장기보존패키지에서 수용할 수 있어야 함. 다만, 기록물 범위에 정의된다면, Original Content/ 또는 Preservation Format/ 폴더에 그렇지 않다면 별도의 폴더(예: Documentation/)를 생성해서 관리되어야 함



<그림 50> 장기보존패키지(NEO3) 생성 2단계

① '원문'은 Original Content/ 폴더에 배치함

② '붙임'도 Original Content/ 폴더에 배치함

③ '행정정보시스템 산출물'도 Original Content/ 폴더에 배치함. 원문이 보존포맷으로 만들어졌다면, Preservation Format/ 폴더는 생성하지 않음. 행정정보시스템이 반드시 보존해야 할 기록물 정의에 포함된다면 Original Content/ 폴더에 배치하고, 반드시 보존해야 할 기록물 정의에 포함되지 않는다면, 최상위에 폴더(예, Documentation/)를 생성하여 저장함

④ '보존포맷'은 Preservation Format/ 폴더에 배치함

⑤ '붙임'이 보존포맷으로 변환된 경우 Preservation Format/ 폴더에 배치함. 원문이 보존포맷으로 만들어졌다면, Preservation Format/ 폴더는 생성하지 않음. Documentation/에 있는 데이터 객체들은 기록물 정의에 포함되지 않는 구성요소이므로 보존포맷으로의 변환 필요 없음

○ (3단계) 기록관리 메타데이터 확정

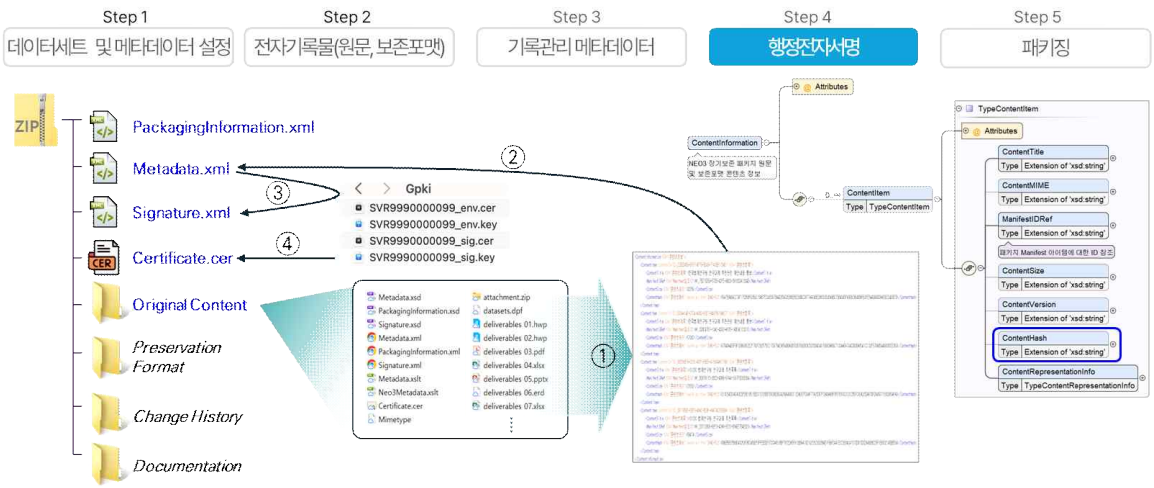
<그림 51> 장기보존패키지(NEO3) 생성 3단계

① 신규 정의한 장기보존 메타데이터로 설정함. 데이터세트용 PreservationDescInfo에는 시스템 및 데이터세트별 메타데이터로 구분됨. 정보시스템 산출물은 'SystemInfoCon → DeliverableCon' XML 요소에 기술하고, 데이터세트별 'DatasetInfoCon → ComponentCon' XML 요소에는 해당 데이터세트에 해당하는 원문, 보존포맷, 붙임 파일들에 대한 정보를 설정함

② Original Content/에 있는 모든 파일에 대해 ContentInformation - ContentItem가 정의된 대로 값(경로, 해시값 등)을 설정함

③ DescriptiveMetadata 요소를 설정함

○ (4단계) 행정전자서명 적용



<그림 52> 장기보존패키지(NEO3) 생성 4단계

① Original Content/에 존재하는 모든 파일에 대한 Hash 값을 생성함

② 생성된 Hash 값들로 Metadata.xml의 Content Information요소를 완성함

③ GPKI 방식으로 행정전자서명을 수행하고, 행정전자서명 값을 생성함

④ 인증서로 Cerificate.cer를 만들고, 행정전자서명 값과 공개키, 유효기간 등을 함께 Signature.xml을 만듦

○ (5단계) 패키징 수행

<그림 53> 장기보존패키지(NEO3) 생성 5단계

① 장기보존패키지 내에 있는 모든 디지털 객체(파일)에 대해 생성날짜, 식별자, 해시값, 경로정보를 목록화하여 PackagingInformation.xml을 생성함

② zip 파일로 패키징을 완료함

바. 장기보존패키지(NEO3) 변경이력(Change History) 관리방안 설계

○ 변경이력 관리 방안 설계의 원칙

- 생성되거나 변환된 모든 장기보존패키지(NEO3)는 보존되어야 하며, 무결성 검증이 가능해야 함 ➔ Data Integrity
- 하나의 NEO3안에는 대량, 대용량 기록물 구성요소들이 존재할 텐데, 중복 저장을 최소화하여 시스템의 효율성을 높임 ➔ Redundancy Minimization

○ 변경이력 관리 방안의 원리 (※ v(n-1)은 이전 버전, v(n)는 새로운 버전)

- v(n) 버전에서는 변경된 구성요소를 포함하여 구성요소 전체 포함
- v(n-1) 버전에는 변경된 구성요소의 직전 버전은 필수 포함

◆ v(n-1) 버전에는 변경된 구성요소의 직전 버전은 필수로 포함되어야 함. 변경되지 않은 구성요소의 직전 버전 정보는 필요에 따라 일부 또는 전부 선택적으로 포함될 수 있음
◆ 최초 생성 버전이거나 파일 대부분이 변경된 버전처럼 대규모 업데이트가 이루어졌을 때는, 해당 버전의 모든 구성요소를 보존하는 것이 복원 시 명확한 기준점을 제공하여 효율적인 복구를 가능하게 함
◆ 이러한 완전한 형태의 버전은 데이터베이스의 '체크포인트(Checkpoint)'나 비디오 압축 기술의 'I-프레임(I-frame)'과 같은 역할을 수행함

❍ 변경이력 관리 방안의 세부 방안

- **[최신본의 최상위 폴더 유지 → 직관적 구조]** 장기보존패키지는 이용의 직관성과 최신성 보장을 위해 최신본을 항상 패키지 루트(/) 에 유지함. 루트에는 당시에 효력을 갖는 메타데이터와 파일들이 존재하며, 관리자는 별도의 탐색 없이도 즉시 최신 상태의 구성과 내용을 확인할 수 있음. 이 원칙은 장기보존 환경에서 빈번히 요구되는 신속한 열람·점검·검증 요청에 대응하기 위한 기본 구조로 기능함

- **[버전별 변경 파일만 보존 가능 → 중복을 최소화한 모든 버전 보존]** 변경이 발생하면 패키지는 Change History/vN/ 폴더를 생성하여 변경되기 전 전체 파일이 아니라 변경된 파일만 원래의 폴더 구조대로 보존함. 버전 번호 N은 변경 이력이 발생할 때마다 1씩 증가함. 이 방식은 변경되지 않은 파일을 중복 저장하지 않기 때문에 저장 공간을 절약할 수 있으며, 변경 시점마다 영향 범위를 명확하게 구분할 수 있다는 장점이 있음. 따라서 각 이력 폴더는 “당시 상태로 복원하는 데 필요한 최소 파일 집합”만을 포함하게 되고, 최신본은 루트 경로에서 일관되게 유지됨. 다만 복원 효율성이나 특정 버전의 특성에 따라, 필요할 경우 변경되지 않은 파일의 일부 또는 전부까지 함께 보존하는 방식도 버전별로 선택할 수 있음

- **[진본확인정보 포함 → 무결성 검증]** 무결성 보장을 위해 Signature.xml과 Certificate.cer을 항상 함께 보존하며, 패키지의 전체 구성요를 확인하기 위해 PackagingInformation.xml을 필수적으로 포함함. 전자는 메타데이터와 패키지 내부 참조의 변조 여부를 검증하는 근거가 되고, 후자는 어떤 파일이 어떤 경로로 존재해야 하는지를 확인하는 기준점으로 기능함. 이 두 구성요소의 상호 보완적 운영은 패키지 수명 전반에 걸친 무결성 검증 체인을 유지하게 함

- **[역방향 병합 복원 → 버전별 전체 구성요소 복원]** 복원은 역방향 병합(Reverse Merge) 원리로 수행함. 먼저 최신본(루트)에서 시작하여 Change History/ 를 버전 번호가 큰 순서대로 역순 탐색하면서 필요한 파일을 덮어써(Overwrite) 감. 이렇게 하면 대상 버전에 이르기까지 각 시점에서 실제로 변경된 파일만이 단계적으로 반영되며, 변경되지 않은 파일은 최신본의 상태를 유지함. 마지막으로, PackagingInformation.xml을 통해 복원 대상 버전에 필요한 전체 구성요소 목록을 확인하고, 목록에 없는 파일들은 삭제함. 이 절차는 저장 효율성을 유지하면서도 목표 시점의 완전한 패키지 구성을 재현하도록 설계됨

- 요약하면, 본 방안은 최신본의 루트 유지로 접근성·관리성을 확보하고, 변경 전 파일만을 보존하는 이력 구조로 저장 효율을 극대화하며, 전자서명·패키징 정보의 상시 포함으로 무결성 검증을 체계화하고, 역방향 병합 복원으로 목표 시점의 패키지를 안정적으로 재현한다는 점에서 장기보존 운용의 실효성과 신뢰성을 동시에 달성함. 변경이력 관리방안은 <표 77>에 요약 정리함

<표 77> 변경이력 관리방안 설명

방안	설명
최신본의 최상위 폴더 유지 (직관적 구조)	· 최신 장기보존패키지는 / 루트 폴더에 유지함
버전별 변경 파일만 선택 보존 (중복 최소화 기반 모든 버전 보존)	· 변경이 발생하면 Change History/vN/ 폴더를 생성하여 해당 시점의 변경되기 직전 파일을 원래의 폴더 구조 그대로 보존함. 필요에 따라 변경되지 않은 파일까지 포함해 저장할 수도 있으며, 버전 번호 N은 변경 이력이 발생할 때마다 1씩 증가함
진본확인정보 포함 (무결성 검증)	· 전자서명 검증을 위한 Signature.xml과 Certificate.cer은 항상 함께 보존함. 전체 구성요소 확인을 위해 PackagingInformation.xml을 보존함
역방향 병합 복원 (버전별 전체 구성요소 보존)	· 복원은 최신본에서 시작하여 Change History/를 역순으로 덮어씌우는(Overwrite) 방식으로 수행함. PackagingInformation.xml의 목록에 존재하는 구성요소들로만 해당 버전을 확정함

○ 변경사항 발생시 변경이력 관리방안의 단계별 절차

- 변경사항 발생시 변경이력 관리방안의 단계별 절차는 <표 78>과 같음

<표 78> 변경이력 관리방안의 단계별 절차

절차		설명
1	변경 감지	· Metadata.xml, PackagingInformation.xml, Certificate.cer, Signature.xml, Original Content/ 폴더, Preservation Format/ 폴더 하위의 파일 중 하나라도 변경이 발생했는지 확인
2	버전 번호 증가 및 폴더 생성	· 기존 Change History/vN/중 가장 큰 버전 번호를 찾아 +1 한 폴더 Change History/vN+1/을 생성
3	변경 전 파일 복사	· 변경되기 직전의 파일만 선택하여 Change History/vN+1/하위에 원래 폴더 및 경로 구조 그대로 복사(필요할 경우 변경되지 않은 파일의 일부 또는 전부 보존가능) ※ Metadata.xml, PackagingInformation.xml, Certificate.cer, Signature.xml도 포함
4	변경된 최신본 반영	· 루트 디렉토리(/)에 변경된 Metadata.xml, Signature.xml, PackagingInformation.xml, Certificate.cer, 구성요소 파일 등을 반영하여 최신 상태로 유지

○ 변경이력 관리방안의 예시를 통한 개념 설명

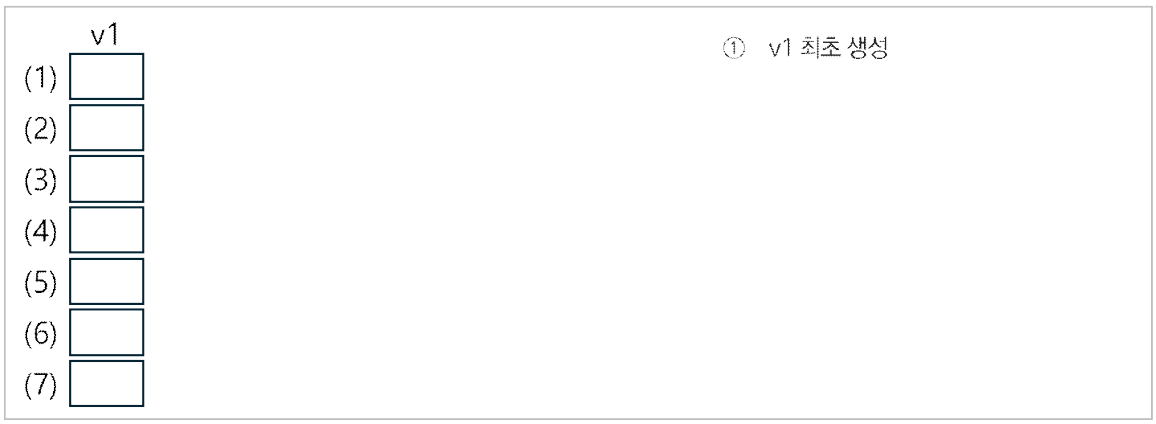
- 예시 설명

· 7개의 구성요소를 가진 v1 버전의 장기보존패키지가 최초에 생성됨. 이때, 각 구성요소는 장기보존패키지를 구성하는 파일에 대응된다고 가정함

· 이후, 1개 또는 그 이상의 구성요소가 변경되면서, v2, v3, v4, v5 버전의 장기보존패키지가 생성되는 과정에서 변경이력이 어떻게 관리되는지를 설명함

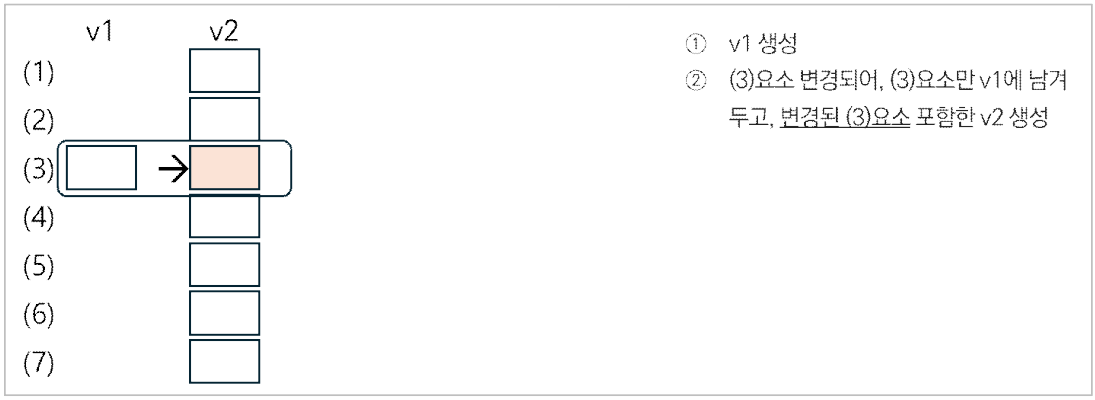
· 본 예시는 중복 저장을 최소화하기 위해, 변경이 발생했을 경우 'Change History/'에는 변경된 구성요소들만 저장하는 방식을 채택함

① v1 최초 생성: 7개의 구성요소를 가진 v1 버전의 장기보존패키지를 생성함. v1의 구성요소들은 장기보존패키지의 '/'에 저장됨



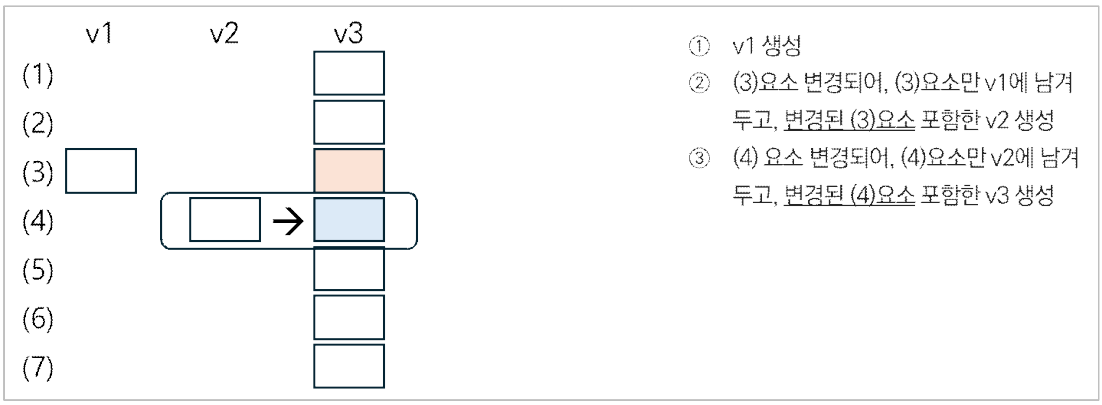
<그림 54> 변경이력 관리방안: v1 최초 생성

② (3)요소 변경: (3)요소만 v1에 남겨 두고, 변경된 (3)요소 포함한 v2를 생성함. v2의 구성요소들은 장기보존패키지의 '/'에 저장되고, v1과 관련된 구성요소는 Change History/v1/ 에 저장됨



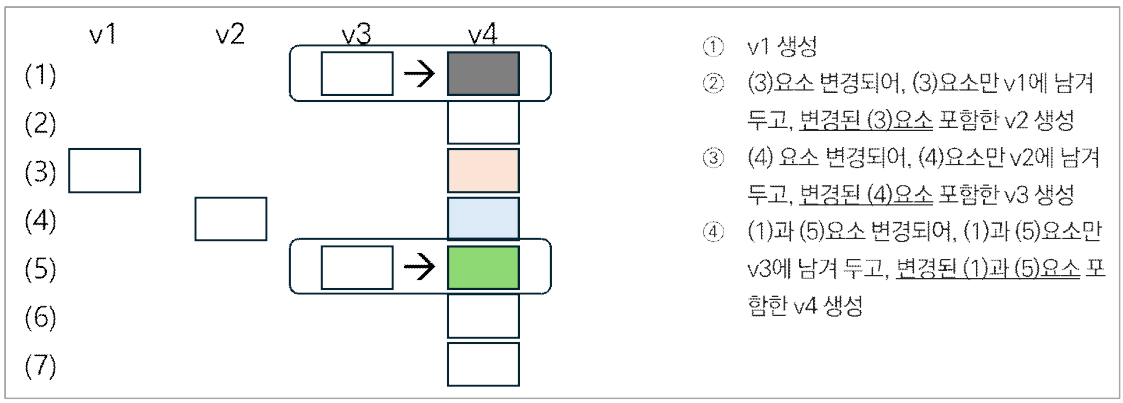
<그림 55> 변경이력 관리방안: (3)요소 변경

③ (4)요소 변경: (4)요소만 v2에 남겨 두고, 변경된 (4)요소 포함한 v3를 생성함. v3의 구성요소들은 장기보존패키지의 '/'에 저장되고, v2와 관련된 구성요소는 Change History/v2/ 에 저장됨



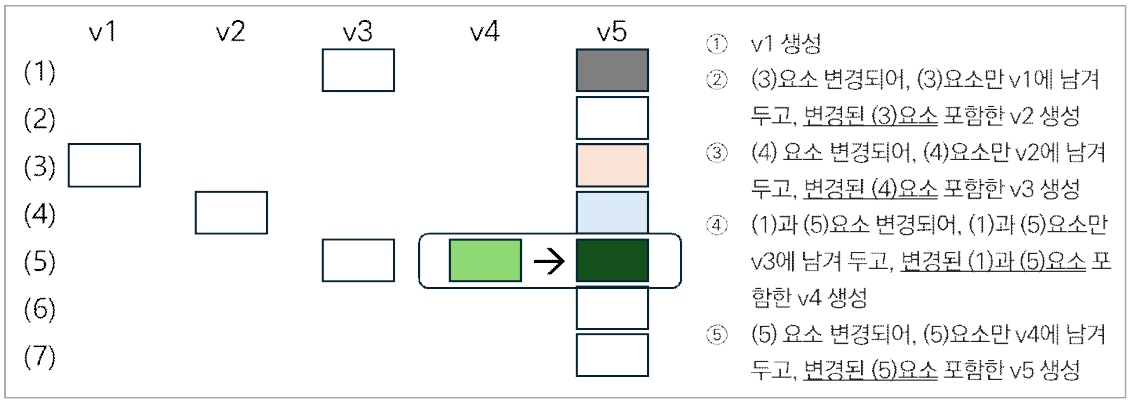
<그림 56> 변경이력 관리방안: (4)요소 변경

④ (1)과 (5)요소 변경: (1), (5)요소만 v3에 남겨 두고, 변경된 (1)과 (5)요소 포함한 v4 생성함. v4의 구성요소들은 장기보존패키지의 '/'에 저장되고, v3와 관련된 구성요소는 Change History/v3/ 에 저장됨



<그림 57> 변경이력 관리방안: (1)과 (5) 요소 변경

⑤ (5)요소 변경: (5)요소만 v4에 남겨 두고, 변경된 (5)요소 포함한 v5를 생성함. v5의 구성요소들은 장기보존패키지의 '/'에 저장되고, v4와 관련된 구성요소는 Change History/v4/ 에 저장됨



<그림 58> 변경이력 관리방안: (5) 요소 변경

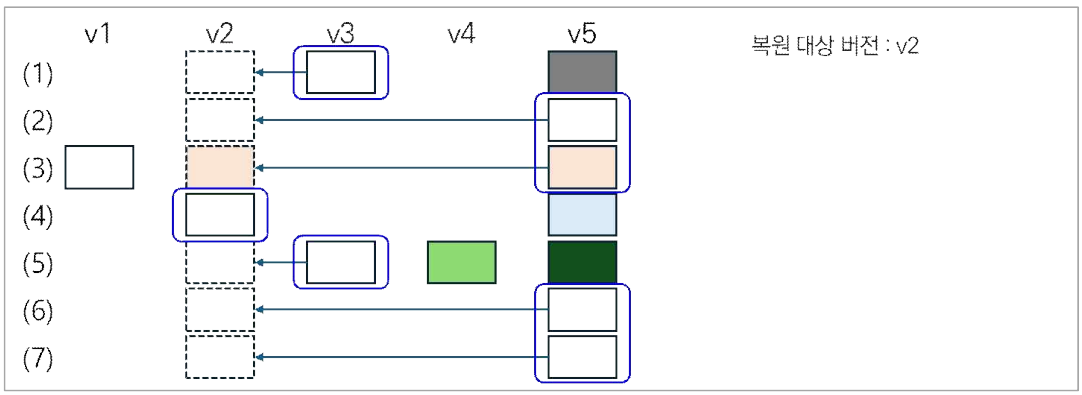
사. 장기보존패키지(NEO3)의 변경이력(Change History) 복원방안 설계

○ 버전별 복원방안 원리 (※ 복원할 버전은 v(p) 이며, 1 < p < n)

· v(n) → v(n-1) → … → v(p) 순서로 진행함

· v(n)의 구성요소에 대해서 v(n-1)의 구성요소들로 덮어쓰기(Overwrite)를 함

· 위 2개의 과정을 v(p)까지 반복함



<그림 59> 변경이력 복원방안 원리에 대한 개념

· 변경이력의 복원방안은 <그림 59>에서처럼 복원 대상 버전(v2) 이후, 다시 변경되기 직전 버전의 구성요소들로 복원하는 방식임. 그래서 v2의 요소를 복원할 때, (1)요소는 v3의 (1) 소로 복원하고, (2)요소는 v5의 요소로, (3)요소는 v5의 요소로, (4)요소는 v2의 요소로, (5)요소는 v3의 요소로, (6)요소는 v5의 요소로, (7)요소도 v5의 요소로 복원함

· 이러한 원리를 실제로 적용하여 구현하기 위해, 최신본에서 출발해 변경이력을 역순으로 확인하고, 해당 버전(v2)에서 다시 변경되기 전의 구성요소들을 덮어써 나감으로써 v2 버전 시점의 완전한 장기보존패키지를 복원할 수 있음

○ 변경이력 복원방안의 단계별 절차

- 변경이력 복원방안의 단계별 절차는 <표 79>와 같음

<표 79> 변경이력 복원방안의 단계별 절차

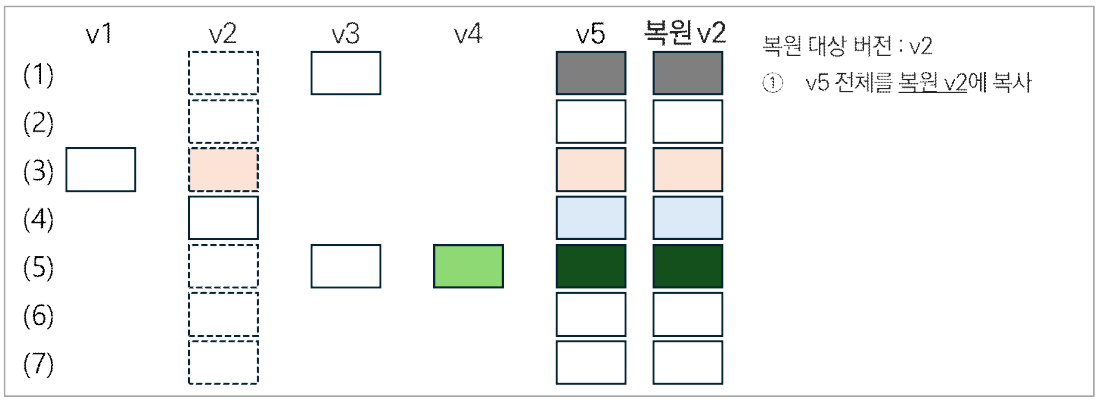
절차		설명
1	복원 대상 버전 지정	· v(1), v(2), …, v(N) 중 복원하려는 장기보존패키지의 버전 번호(vP, 1≤P<N)를 선택한다.
2	최신본기준 복사	· 현재 루트(/)에 위치한 최신본 전체를 특정 폴더를 생성하여 해당 폴더(restore_vP/)로 복사함. 이 복제본을 기반으로 복원을 진행함
3	패키징 정보 기반 역순 병합	· Change History/vN/부터 Change History/vP/까지 순차적으로 역순 탐색하며, 해당 버전에 저장된 파일을 복제본에 복사함. 폴더 및 파일 경로는 원본과 동일하게 유지되어 있으므로 자동 병합이 가능함. · 이때, Change History/vP/PackagingInformation.xml에 정의되지 않은 파일은 제외하고 복사함
4	구성요소 확인	· Change History/vP/PackagingInformation.xml에 명시된 파일들의 목록을 이용하여 restore_vP/에 존재하는지 확인하고, 나머지 파일들을 삭제함 · vP 버전 이후 보존 과정(예: 보존포맷 변환, 재현도구 변경 등)에서 장기보존패키지에 추가로 생성된 파일이 있을 수 있음. 3절차에서 이러한 파일들이 vP까지 함께 복사되므로, 복원 시에는 해당 파일들을 장기보존패키지에서 제거해야 함
5	무결성 검증	· Change History/vP/PackagingInformation.xml에 정의된 해시값과 각 파일의 실제 해시를 비교하여 복원 결과의 무결성을 검증함

○ 변경이력 복원방안의 예시를 통한 개념 설명

- 예시 설명

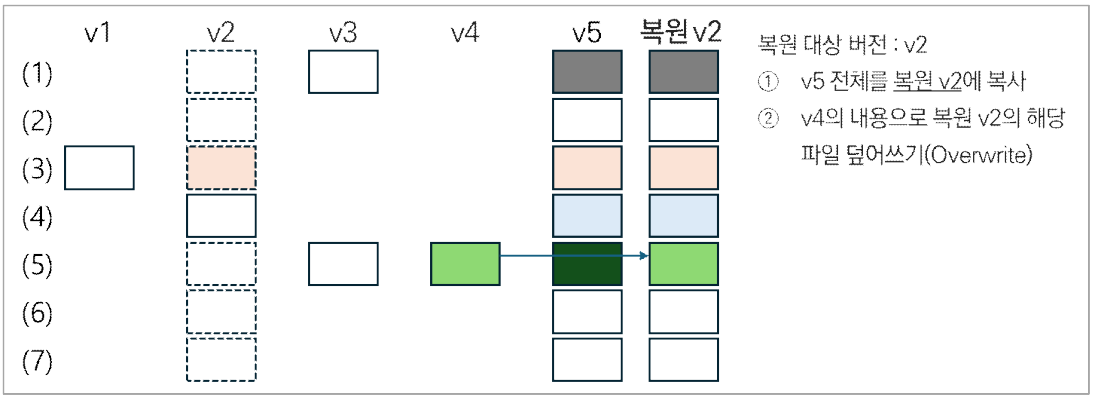
· 복원대상 버전이 v2로 설정하고, 7개의 구성요소를 가진 v1 버전의 장기보존패키지가 최초에 생성됨. 이후, 1개 또는 그 이상의 구성요소가 변경되면서, v2, v3, v4, v5 버전의 장기보존패키지가 생성되는 과정에서 변경이력이 어떻게 관리되는지를 설명함

① 최신본 v5에서 시작: v5 전체를 '복원 v2'에 복사함. Change History/v5/ 폴더에 있는 파일들을 기존의 폴더 및 파일 경로 구조를 그대로 유지한 상태로 restore_vP/ 폴더에 전체 복사



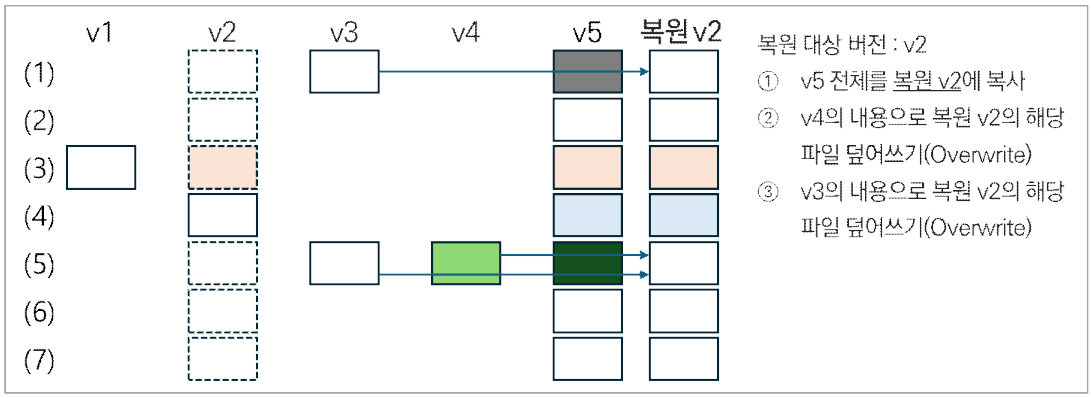
<그림 60> 변경이력 복원방안: 최신본 v5 복사

② v4 버전의 내용 복원: v4의 내용으로 '복원 v2'의 해당 요소들을 덮어쓰기(Overwrite). Change History/v4/ 폴더에 있는 파일들을 원래의 폴더 및 파일 경로 구조를 그대로 유지한 상태로 restore_vP/ 폴더에 덮어씀



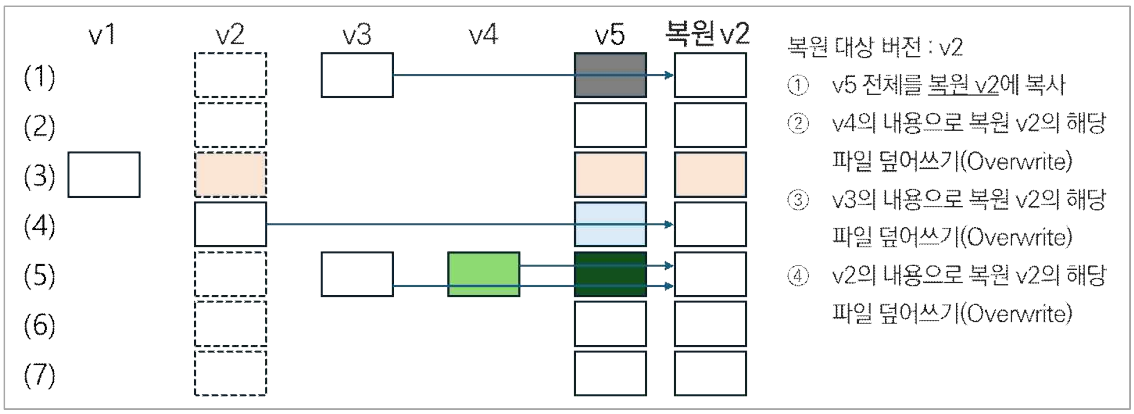
<그림 61> 최신본 v4 버전의 내용 복원

③ v3 버전의 내용 복원: v3의 내용으로 '복원 v2'의 해당 요소들을 덮어쓰기. Change History/v3/ 폴더에 있는 파일들을 원래의 폴더 및 파일 경로 구조를 그대로 유지한 상태로 restore_vP/ 폴더에 덮어씀



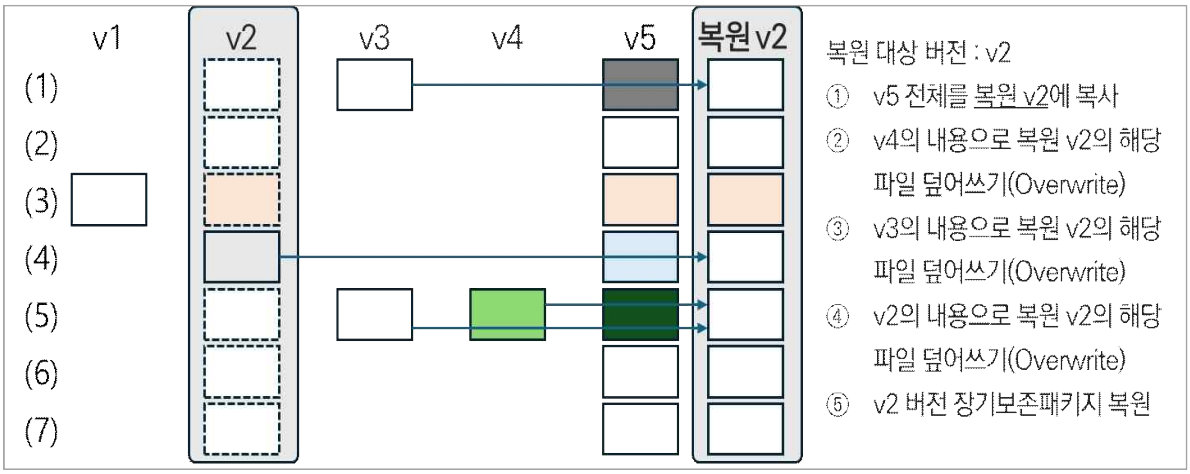
<그림 62> 변경이력 복원방안: 최신본 v3 버전의 내용 복원

④ v2 버전의 내용 복원: v2의 내용으로 '복원 v2'의 해당 요소들을 덮어쓰기. Change History/v2/ 폴더에 있는 파일들을 원래의 폴더 및 파일 경로 구조를 그대로 유지한 상태로 restore_vP/ 폴더에 덮어씀



<그림 63> 변경이력 복원방안: 최신본 v2 버전의 내용 복원

⑤ v2 버전 장기보존패키지 생성: restore_vP/PackagingInformation.xml의 파일들에 대한 목록 정보에 있는 파일들이 restore_vP/ 내에 있는 모두 존재하는지 확인하고, 목록 정보에는 없지만 restore_vP/에는 존재하는 파일들은 제거함. 복원 대상인 v2 버전의 장기보존패키지의 모든 내용을 완전히 복원함



<그림 64> 변경이력 복원방안: 최신본 v2 버전의 장기보존패키지 복원

3.6 재현 정보를 고려한 보존포맷 및 장기보존패키지 구조 및 설정 방안

가. 재현 정보의 필요성과 개념

○ 재현 정보(representation information)는 미래의 이용자가 기록을 이해·해석·재현할 수 있도록 지원하는 기술적·맥락적 정보를 의미함. 특히 데이터세트(DB형) 유형 기록물은 원시 데이터만으로는 구조적 의미, 테이블 간 관계, 제약조건, 애플리케이션 의존성 등을 파악하기 어려우므로, 재현 정보는 기록의 내용(content)과 구조(structure)를 실질적으로 복원하는 데 필수적인 요소임. 또한 전자기록물의 필수보존속성(Significant Properties)에 '기능(behavior)'이 별도의 범주로 포함된 것도 이러한 맥락과 관련됨. 전자기록물은 기록 자체의 의미뿐 아니라 시스템 환경이나 이용자의 조작에 따라 변화·반응하는 기능적 특성까지 보존해야 하기 때문임.

○ 이러한 특성을 고려하면, 데이터세트(DB형) 유형 기록물의 보존에서는 해당 데이터세트가 어떤 목적을 위해 생성·운영되었는지, 그리고 그 목적을 충족하기 위해 어떠한 재현 정보가 필요한지를 함께 검토해야 함. 즉, 데이터세트의 본래 기능과 활용 방식을 이해하고, 이를 뒷받침하는 재현 정보를 함께 보존할 때 비로소 기록의 의미와 맥락을 온전히 재현할 수 있음

나. 재현 정보를 고려한 보존포맷 및 장기보존패키지의 구조 및 설정 방안

○ 재현 정보 관련 보존포맷의 구조 및 설정 방안

- 오픈포맷 기반 보존포맷에서는 데이터세트(DB형) 기록물의 구조와 내용을 기술하는 metadata.xml과, 실제 재현을 수행하는 지원도구를 체계적으로 연결할 수 있도록 폴더 구조(representation/)와 요소 구성을 정의함

- metadata.xml의 <table> 요소 내 재현 정보 명세: 각 테이블을 기술하는 <table> 요소는 해당 테

이블의 구조적 속성과 함께, 재현에 필요한 지원도구를 명시한다. 이때 다음 정보가 포함됨

· <representationInfo> 요소: 재현 정보를 명시할 수 있는 요소

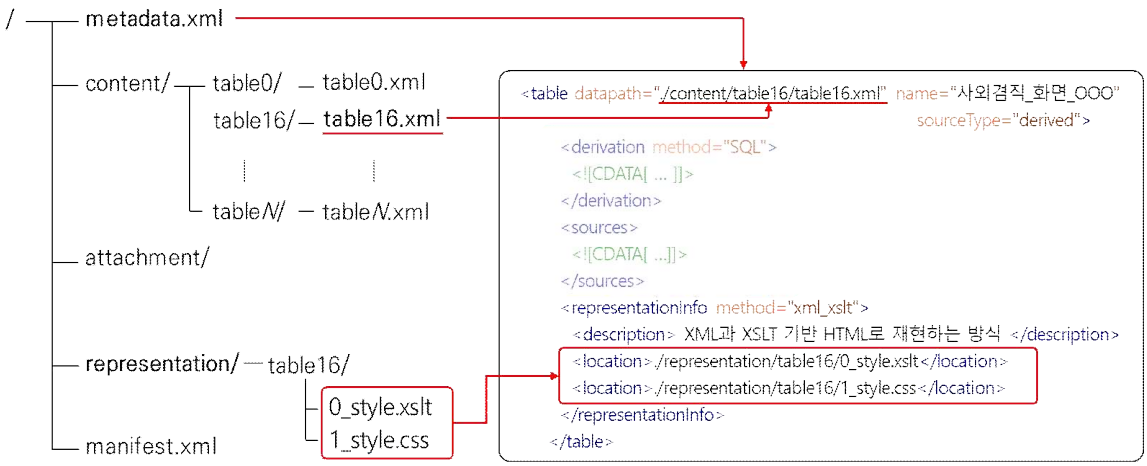
· method 속성: 테이블 재현 방식(XSLT 처리 방식 등)

· <description> 요소: 표시 형태, 적용 목적 등 재현 방식에 대한 설명

· <location> 요소: 관련 XSL 파일의 상대 경로(URI)

- representation/ 폴더의 지원도구 저장 구조: 재현에 사용되는 XSL, CSS 등 지원도구는 보존포맷 내부의 representation/ 폴더에 저장하며, 폴더 내부의 구체적 구조는 기록의 특성과 필요에 따라 자유롭게 구성할 있음

- <그림 65>는 XML/XSLT 기반 재현도구를 보존포맷 내부에 포함시키는 방법을 예시로 보여줌. metadata.xml의 <table> 요소 중 table16.xml에 해당하는 부분을 살펴보면, 이 테이블은 XML/XSLT 방식으로 재현되며, 필요한 재현 도구는 representation/table16/ 폴더에 저장된 0_style.xslt과 1_style.css라는 점을 확인할 수 있음



<그림 65> 재현정보 관련 보존포맷의 구조 및 설정 방안 예시(XML_XSLT)

○ 재현 정보 관련 장기보존패키지의 구조 및 설정 방안

- 재현도구는 장기보존패키지의 Representation Information/ 폴더에 저장되며, 이 폴더에는 기록을 재현하는 데 필요한 다양한 지원도구가 포함됨. Representation Information/ 폴더의 구조는 기관의 정책과 기록의 성격에 따라 유연하게 구성할 수 있음

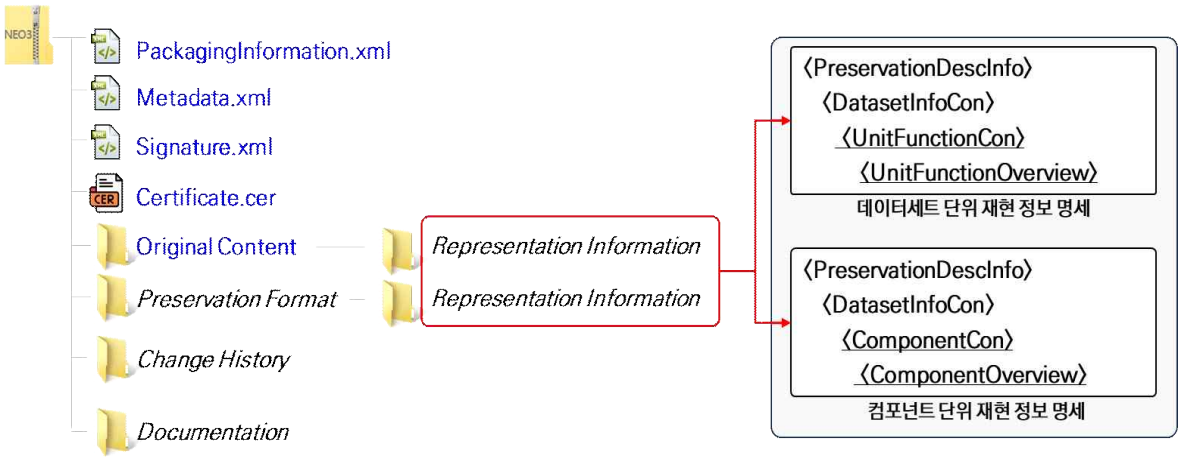
- 예를 들어, SIARD Suite 실행파일 또는 설치 파일, 해당 도구의 버전 정보와 의존 라이브러리, 그리고 필요시 활용 매뉴얼이나 설정 파일을 함께 보관할 수 있음. 또한 재현 과정에서 추가적인 처리가 필요한 경우를 대비하여 SQL 스크립트나 기술 문서와 같은 보조 자료 등도 포함할 수도 있음

- 장기보존패키지에서는 재현 정보를 데이터세트 별로 정의하는 것을 원칙으로 함. 이에 따라 metadata.xml에서는 재현 정보를 명세할 수 있는 두 가지 위치를 <그림 66>과 같이 제시할 수 있음

· 첫째, 데이터세트의 기능적 목적과 역할에 따라 재현 방안을 설명해야 하는 경우, <PreservationDescInfo> → <DatasetInfoCon> → <UnitFunctionCon> → <UnitFunctionOverview> 요소에서 재현 정보를 기술할 수 있음

· 둘째, 데이터세트 내부의 구성요소(컴포넌트)별로 서로 다른 재현도구가 필요한 경우에는 <PreservationDescInfo> → <DatasetInfoCon> → <ComponentCon> →

<ComponentOverview> 요소에서 컴포넌트 단위의 재현 정보를 각각 명세할 수 있음. 이를 통해 데이터세트 전체 또는 개별 파일 단위에서 요구되는 재현 요구사항을 체계적으로 표현할 수 있음



<그림 66> 재현정보 관련 장기보존패키지의 구조 및 설정

3.7 데이터세트 유형 보존포맷 및 장기보존패키지 생성을 위한 테스트베드 구축 및 실데이터 기반 검증

가. 실데이터 기반 데이터베이스 생성 및 테스트베드 구축

○ 검증 대상 시스템 구성

- 개요

· 실데이터 기반 장기보존패키지 검증을 위해 전북대학교에서 운영하는 내부 행정정보시스템인 '오아시스 3.0(OASIS 3.0)'의 개인인사기본 메뉴를 기준으로 데이터베이스를 생성함

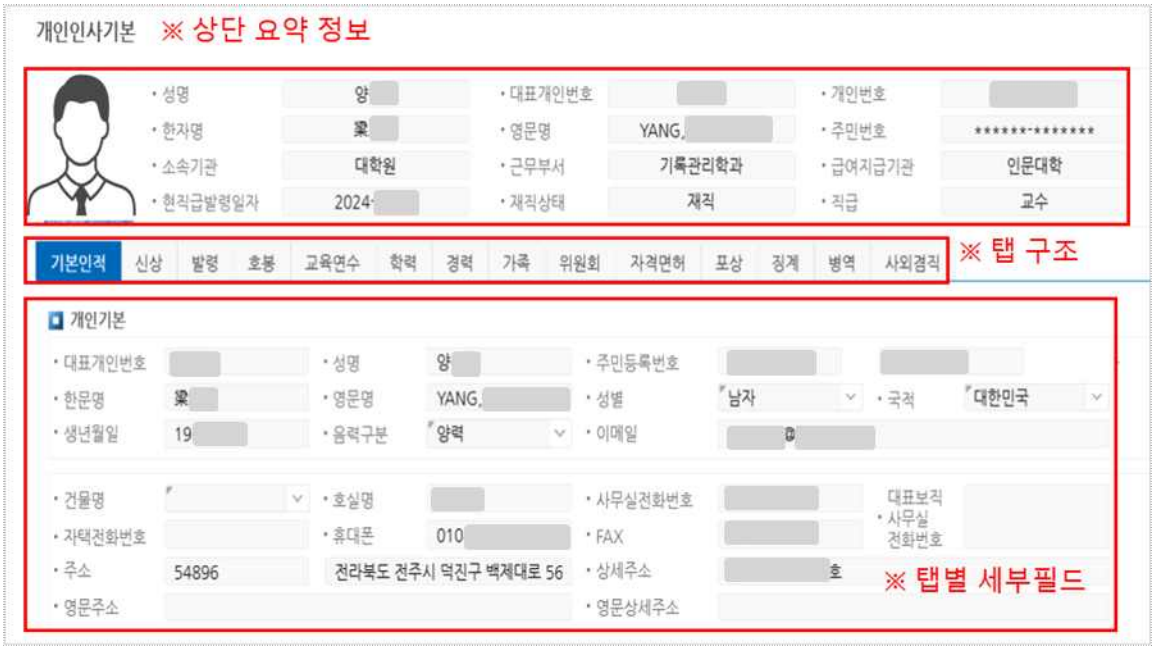
· 오아시스 3.0은 교직원 및 학생 개개인의 인사, 복무, 계약, 포상, 징계, 교육 이력 등과 같은 다양한 행정정보를 통합 관리하고 웹 기반 인터페이스를 통해 사용자 접근성을 제공함

- 구성 및 특징

· 개인인사기본 메뉴는 상단 요약 정보와 하단 탭 형식의 상세 정보 영역으로 나뉘며, 한 명의 인물에 대한 이력을 한 화면에서 확인할 수 있도록 구성되어 있음(<그림 67> 참조)

· 개인인사기본 메뉴의 탭 구조는 '기본인적', '경력', '호봉', '발령', '포상', '징계', '자격'과 같은 속성별 정보를 세분화하여 제공함

· 사용자는 각 탭을 클릭하여 관련 행정정보를 체계적으로 탐색할 수 있으며, 각 탭은 복수의 세부 필드로 구성됨



<그림 67> 오아시스 3.0 개인인사기본 메뉴 화면

❍ 검증 대상 시스템 데이터베이스 생성

- 검증 대상 시스템 기반 데이터베이스는 MySQL 데이터베이스를 기반으로 한 관계형 데이터베이스(RDBMS) 구조로 가정함

- 데이터베이스 호환 버전

· MySQL 버전: MariaDB 10.4.32

· 데이터베이스 개발 도구: Version 8.0.38 build 4270059 CE (64bits)

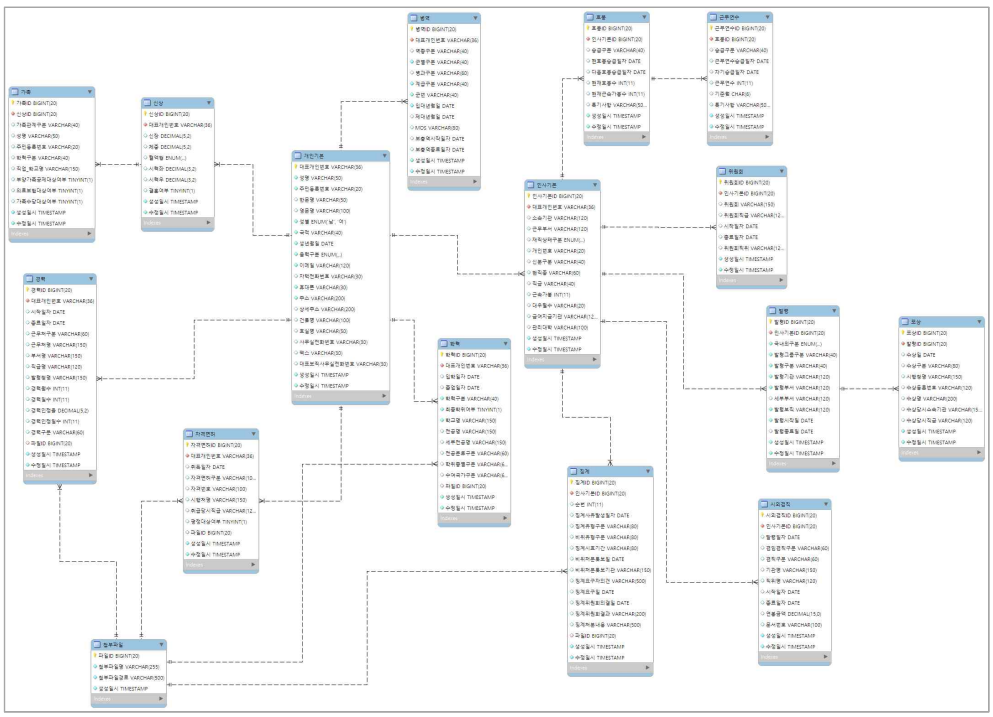
- 데이터베이스 구조

· 데이터베이스는 <표 80>과 같이 개인기본, 인사기본, 경력, 근무연수 등 총 16개의 테이블로 구성함

<표 80> 데이터베이스 내 테이블 데이터 건수

순번	테이블명	데이터 수(건)
1	개인기본	20
2	가족	27
3	호봉	159
4	경력	19
5	근무연수	72
6	발령	30
7	병역	22
8	시외겸직	8
9	신상	20
10	위원회	4
11	인사기본	20
12	자격면허	13
13	징계	2
14	첨부파일	16
15	포상	13
16	학력	48
총 데이터 건수		493

· 검증 대상 시스템을 기반으로 생성한 관계형 데이터베이스의 구조는 <그림 68>와 같음



<그림 68> 검증 대상 시스템 기반 데이터베이스 ERD

- 데이터베이스 테이블 설계

· 개인기본 테이블 명세

<표 81> 개인기본 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
대표개인번호	VARCHAR(36)	PK	NOT NULL		
성명	VARCHAR(50)		NOT NULL		
주민등록번호	VARCHAR(20)		NOT NULL		
한문명	VARCHAR(50)				NULL
영문명	VARCHAR(100)				NULL
성별	ENUM(“남”,“여”)		NOT NULL		
국적	VARCHAR(40)		NOT NULL		
생년월일	DATE		NOT NULL		
음력구분	ENUM(“양력”,“음력”)		NOT NULL		
이메일	VARCHAR(120)		NOT NULL		
자택전화번호	VARCHAR(30)				NULL
휴대폰	VARCHAR(30)		NOT NULL		
주소	VARCHAR(200)		NOT NULL		
상세주소	VARCHAR(200)				NULL
건물명	VARCHAR(100)				NULL
호실명	VARCHAR(50)				NULL
사무실전화번호	VARCHAR(30)				NULL
팩스	VARCHAR(30)				NULL
대표보직사무실전화번호	VARCHAR(30)				NULL
생성일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP()
수정일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP() ON UPDATE CURRENT_TIMESTAMP()

· 가족 테이블 명세

<표 82> 가족 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
가족ID	BIGINT(20)	PK	NOT NULL		
신상ID	BIGINT(20)		NOT NULL	FOREIGN KEY (‘신상ID’) REFERENCES ‘신상’(‘신상ID’) ON DELETE CASCADE	
가족관계구분	VARCHAR(40)				NULL
성명	VARCHAR(50)				NULL
주민등록번호	VARCHAR(20)				NULL
학력구분	VARCHAR(40)				NULL
직업_학교명	VARCHAR(150)				NULL
부양가족공제대상여부	TINYINT(1)				NULL
의료보험대상여부	TINYINT(1)				NULL
가족수당대상여부	TINYINT(1)				NULL
생성일시	TIMESTAMP				CURRENT_TIMESTAMP()
수정일시	TIMESTAMP				CURRENT_TIMESTAMP() ON UPDATE CURRENT_TIMESTAMP()

· 호봉 테이블 명세

<표 83> 호봉 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
호봉ID	BIGINT(20)	PK	NOT NULL		
인사기본ID	BIGINT(20)		NOT NULL	FOREIGN KEY (‘인사기본ID’) REFERENCES ‘인사기본’ (‘인사기본ID’) ON DELETE CASCADE	
승급구분	VARCHAR(40)				NULL
현호봉승급일자	DATE				NULL
다음호봉승급일자	DATE				NULL
현재호봉수	INT(11)				NULL
현재근속가봉수	INT(11)				NULL
특기사항	VARCHAR(500)				NULL
생성일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP()
수정일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP() ON UPDATE CURRENT_TIMESTAMP()

· 경력 테이블 명세

<표 84> 경력 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
경력ID	BIGINT(20)	PK	NOT NULL		
대표개인번호	VARCHAR(36)		NOT NULL	FOREIGN KEY (‘대표개인번호’) REFERENCES ‘개인기본’ (‘대표개인번호’) ON DELETE CASCADE	
시작일자	DATE				NULL
종료일자	DATE				NULL
근무처구분	VARCHAR(60)				NULL
근무처명	VARCHAR(150)				NULL
부서명	VARCHAR(150)				NULL
직급명	VARCHAR(120)				NULL
발령청명	VARCHAR(150)				NULL
경력월수	INT(11)				NULL
경력일수	INT(11)				NULL
경력인정율	DECIMAL(5,2)				NULL
경력인정일수	INT(11)				NULL
경력구분	VARCHAR(60)				NULL
파일ID	BIGINT(20)			FOREIGN KEY (‘파일ID’) REFERENCES ‘첨부파일’ (‘파일ID’)	NULL
생성일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP()
수정일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP() ON UPDATE CURRENT_TIMESTAMP()

· 근무연수 테이블 명세

<표 85> 근무연수 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
근무연수ID	BIGINT(20)	PK	NOT NULL		
호봉ID	BIGINT(20)		NOT NULL	FOREIGN KEY (‘호봉ID’) REFERENCES ‘호봉’ (‘호봉ID’) ON DELETE CASCADE	
승급구분	VARCHAR(40)				NULL
근무연수승급일자	DATE				NULL
차기승급일자	DATE				NULL
근무연수	INT(11)				NULL
기준월	CHAR(6)				NULL
특기사항	VARCHAR(500)				NULL
생성일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP()
수정일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP() ON UPDATE CURRENT_TIMESTAMP()

· 발령 테이블 명세

<표 86> 발령 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
발령ID	BIGINT(20)	PK	NOT NULL		
인사기본ID	BIGINT(20)		NOT NULL	FOREIGN KEY (‘인사기본ID’) REFERENCES ‘인사기본’ (‘인사기본ID’) ON DELETE CASCADE	
국내외구분	ENUM(“국내”,“국외”)		NOT NULL		
발령그룹구분	VARCHAR(40)		NOT NULL		
발령구분	VARCHAR(40)		NOT NULL		
발령기관	VARCHAR(120)		NOT NULL		
발령부서	VARCHAR(120)		NOT NULL		
세부부서	VARCHAR(120)				NULL
발령보직	VARCHAR(120)		NOT NULL		
발령시작일	DATE		NOT NULL		
발령종료일	DATE				NULL
생성일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP()
수정일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP() ON UPDATE CURRENT_TIMESTAMP()

· 병역 테이블 명세

<표 87> 병역 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
병역ID	BIGINT(20)	PK	NOT NULL		
대표개인번호	VARCHAR(36)		NOT NULL	FOREIGN KEY (‘대표개인번호’) REFERENCES ‘개인기본’ (‘대표개인번호’) ON DELETE CASCADE	
역종구분	VARCHAR(40)				NULL
군별구분	VARCHAR(40)		NOT NULL		
병과구분	VARCHAR(60)				NULL
계급구분	VARCHAR(40)		NOT NULL		
군번	VARCHAR(40)		NOT NULL		
입대년월일	DATE		NOT NULL		
제대년월일	DATE				NULL
MOS	VARCHAR(80)				NULL
보충역시작일자	DATE				NULL
보충역종료일자	DATE				NULL
생성일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP()
수정일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP() ON UPDATE CURRENT_TIMESTAMP()

· 시외겸직 테이블 명세

<표 88> 시외겸직 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
시외겸직ID	BIGINT(20)	PK	NOT NULL		
인사기본ID	BIGINT(20)		NOT NULL	FOREIGN KEY (‘인사기본ID’) REFERENCES ‘인사기본’ (‘인사기본ID’) ON DELETE CASCADE	
발령일자	DATE				NULL
겸임겸직구분	VARCHAR(60)				NULL
겸직구분	VARCHAR(60)				NULL
기관명	VARCHAR(150)				NULL
직위명	VARCHAR(120)				NULL
시작일자	DATE				NULL
종료일자	DATE				NULL
연봉금액	DECIMAL(15,0)				NULL
파일ID	BIGINT(20)			FOREIGN KEY (‘파일ID’) REFERENCES ‘첨부파일’ (‘파일ID’) ON DELETE CASCADE	NULL
생성일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP()
수정일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP() ON UPDATE CURRENT_TIMESTAMP()

· 신상 테이블 명세

<표 89> 신상 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
신상ID	BIGINT(20)	PK	NOT NULL		
대표개인번호	VARCHAR(36)		NOT NULL	FOREIGN KEY (‘대표개인번호’) REFERENCES ‘개인기본’ (‘대표개인번호’) ON DELETE CASCADE	
신장	DECIMAL(5,2)				NULL
체중	DECIMAL(5,2)				NULL
혈액형	ENUM(‘A’,’B’,’O’,’AB’)				NULL
시력좌	DECIMAL(3,2)				NULL
시력우	DECIMAL(3,2)				NULL
결혼여부	TIMYINT(1)				NULL
생성일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP()
수정일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP() ON UPDATE CURRENT_TIMESTAMP()

· 위원회 테이블 명세

<표 90> 위원회 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
위원회ID	BIGINT(20)	PK	NOT NULL		
인사기본ID	BIGINT(20)		NOT NULL	FOREIGN KEY (‘인사기본ID’) REFERENCES ‘인사기본’ (‘인사기본ID’) ON DELETE CASCADE	
위원회	VARCHAR(150)				NULL
위원회직급	VARCHAR(120)				NULL
시작일자	DATE				NULL
종료일자	DATE				NULL
위원회직위	VARCHAR(120)				NULL
생성일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP()
수정일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP() ON UPDATE CURRENT_TIMESTAMP()

· 인사기본 테이블 명세

<표 91> 인사기본 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
인사기본ID	BIGINT(20)	PK	NOT NULL		
대표개인번호	VARCHAR(36)		NOT NULL	FOREIGN KEY (‘대표개인번호’) REFERENCES ‘개인기본’ (‘대표개인번호’) ON DELETE CASCADE	
소속기관	VARCHAR(120)				NULL
근무부서	VARCHAR(120)				NULL
재직상태구분	ENUM('재직', '휴직', '파견', '겸직', '퇴직', '정년퇴직')				‘재직’
개인번호	VARCHAR(20)				NULL
신분구분	VARCHAR(40)				NULL
현직종	VARCHAR(60)				NULL
직급	VARCHAR(40)				NULL
근속가봉	INT(11)				NULL
대우필수	VARCHAR(20)				NULL
급여지급기관	VARCHAR(120)				NULL
관리대학	VARCHAR(100)				NULL
생성일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP()
수정일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP() ON UPDATE CURRENT_TIMESTAMP()

· 자격면허 테이블 명세

<표 92> 자격면허 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
자격면허ID	BIGINT(20)	PK	NOT NULL		
대표개인번호	VARCHAR(36)		NOT NULL	FOREIGN KEY (‘대표개인번호’) REFERENCES ‘개인기본’ (‘대표개인번호’) ON DELETE CASCADE	
취득일자	DATE				NULL
자격면허구분	VARCHAR(100)				NULL
자격번호	VARCHAR(100)				NULL
시행처명	VARCHAR(150)				NULL
취급당시직급	VARCHAR(120)				NULL
평정대상여부	TINYINT(1)				NULL
파일ID	BIGINT(20)			FOREIGN KEY (‘파일ID’) REFERENCES ‘첨부파일’ (‘파일ID’) ON DELETE CASCADE	NULL
생성일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP()
수정일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP() ON UPDATE CURRENT_TIMESTAMP()

· 징계 테이블 명세

<표 93> 징계 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
징계ID	BIGINT(20)	PK	NOT NULL		
인사기본ID	BIGINT(20)		NOT NULL	FOREIGN KEY (‘인사기본ID’) REFERENCES ‘인사기본’ (‘인사기본ID’) ON DELETE CASCADE	
순번	INT(11)				NULL
징계사유발생일자	DATE				NULL
징계유형구분	VARCHAR(80)				NULL
비위유형구분	VARCHAR(80)				NULL
징계시효기간	VARCHAR(80)				NULL
비위처분통보일	DATE				NULL
비위처분통보기관	VARCHAR(150)				NULL
징계요구자의견	VARCHAR(500)				NULL
징계요구일	DATE				NULL
징계위원회의결일	DATE				NULL
징계위원회결과	VARCHAR(200)				NULL
징계처분내용	VARCHAR(500)				NULL
파일ID	BIGINT(20)			FOREIGN KEY (‘파일ID’) REFERENCES ‘첨부파일’ (‘파일ID’)	NULL
생성일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP()
수정일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP() ON UPDATE CURRENT_TIMESTAMP()

· 첨부파일 테이블 명세

<표 94> 첨부파일 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
파일ID	BIGINT(20)		NOT NULL		
첨부파일명	VARCHAR(255)		NOT NULL		
첨부파일경로	VARCHAR(500)		NOT NULL		
생성일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP()
파일UUID	CHAR(36)	PK	NOT NULL		

· 포상 테이블 명세

<표 95 포상> 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
포상ID	BIGINT(20)	PK	NOT NULL		
발령ID	BIGINT(20)		NOT NULL	FOREIGN KEY (‘발령ID’) REFERENCES ‘발령’ (‘발령ID’) ON DELETE CASCADE	
수상일	DATE				NULL
수상구분	VARCHAR(80)				NULL
시행청명	VARCHAR(150)				NULL
수상등록번호	VARCHAR(120)				NULL
수상명	VARCHAR(200)				NULL
수상당시소속기관	VARCHAR(150)				NULL
수상당시직급	VARCHAR(120)				NULL
생성일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP()
수정일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP() ON UPDATE CURRENT_TIMESTAMP()

· 학력 테이블 명세

<표 96> 학력 테이블 상세내역

컬럼명	데이터타입	PK 여부	NOT NULL 여부	FK 여부	기본값
학력ID	BIGINT(20)	PK	NOT NULL		
대표개인번호	VARCHAR(36)		NOT NULL	FOREIGN KEY (‘대표개인번호’) REFERENCES ‘개인기본’ (‘대표개인번호’) ON DELETE CASCADE	
입학일자	DATE				NULL
졸업일자	DATE				NULL
학력구분	VARCHAR(40)		NOT NULL		
최종학위여부	TINYINT(1)		NOT NULL		
학교명	VARCHAR(150)		NOT NULL		
전공명	VARCHAR(150)				NULL
세부전공명	VARCHAR(150)				NULL
전공분류구분	VARCHAR(60)				NULL
학위종별구분	VARCHAR(60)				NULL
수여국가구분	VARCHAR(60)				NULL
파일ID	BIGINT(20)			FOREIGN KEY (‘파일ID’) REFERENCES ‘첨부파일’ (‘파일ID’)	NULL
생성일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP()
수정일시	TIMESTAMP		NOT NULL		CURRENT_TIMESTAMP() ON UPDATE CURRENT_TIMESTAMP()

- 데이터베이스 객체(Database Objects) 데이터 추가

· 데이터베이스 객체는 데이터베이스에서 애플리케이션 또는 사용자별 데이터를 저장·관리·표시하기 위한 기본적인 단위로, 데이터베이스 스키마의 구성요소로서 데이터베이스의 구조, 기능 등을 정의함

· 검증 대상 시스템 기반 데이터베이스 생성 시 테이블 및 데이터 입력과 같은 구조적 객체 이외의 Trigger, View, Function, Event 등의 비구조적인 객체를 포함하기 위해 <표 97>과 같은 데이터베이스 객체 데이터를 추가함

<표 97> 데이터베이스 추가 객체 정보

구분		객체명	객체 상세
Trigger		trg_첨부파일_bi	· 첨부파일이 새로 등록될 때 파일 경로(첨부파일경로)를 자동으로 지정된 경로로 설정
		trg_첨부파일_bu	· 첨부파일명이 변경될 때마다 파일 경로를 다시 자동 갱신
Function	Procedure	sp_호봉_다음승급일_계산	· 호봉 테이블의 현호봉승급일자 기준으로 다음호봉승급일자를 1년 뒤로 자동 계산·갱신
	Routine	fn_나이계산	· 생년월일을 기준으로 현재 나이(만 나이)를 반환
Event		ev_호봉_월간보정	· 매달 1회 sp_호봉_다음승급일_계산() 프로시저를 자동 실행하여 호봉 데이터 정기 보정 수행
View		vw_재직자현황	· 인사기본 테이블, 개인기본 테이블을 JOIN하여 현재 재직 중인 인원 목록 제공
		vw_최근발령	· 발령 테이블에서 종료되지 않은 최신 발령 이력 조회
Index			· 속도 향상 및 데이터 최적화
User		기록연구사	· 데이터베이스 사용자 역할 생성 및 권한 부여 · 로그인 시 자동으로 역할이 활성화되도록 설정
		업무담당자 A, B	
		시스템담당자 C, D	

❍ 장기보존패키지(NEO3)의 구성요소 설정 방안

- 개요

· 검증 대상 시스템 기반 데이터베이스에 Python 기반 SQL-XML 추출 및 변환 코드를 적용하여 XML 데이터를 추출함. 이를 '테이블중심 XML 추출 방식'과 '주요개체중심 XML 추출 방식'으로 구분하여 제시함

· 추출한 XML 파일은 XSLT 파일과 결합하여 오아시스 3.0 시스템의 '개인인사기본' 메뉴 화면과 동일한 구조와 형식의 HTML 파일로 재현함. 이때 XML 파일은 장기보존패키지(NEO3) 내 보존포맷으로, XSLT 파일과 HTML 파일은 재현 정보(Representation Information)로 구성됨

- 데이터베이스 추출 및 XML 파일 변환 과정

· 검증 대상 정보시스템은 MySQL 기반 관계형 데이터베이스 환경으로 구성되어 있으며, Python 코드를 통해 MySQL 서버에 접속하여 데이터를 조회함

· Python의 'pymysql' 라이브러리를 사용하여 데이터베이스 연결을 설정하고, 'SHOW TABLES', 'SELECT * FROM 테이블명' 등의 쿼리를 통해 테이블 구조 및 전체 데이터를 조회함

```python
import pymysql
import xml.etree.ElementTree as ET
import os

# 현재 폴더 기준 경로 설정 (스크립트가 최상위 project 폴더에 있음)
BASE_DIR = os.getcwd()
XML_DIR = os.path.join(BASE_DIR, "xml")
HTML_DIR = os.path.join(BASE_DIR, "html")
XSLT_DIR = os.path.join(BASE_DIR, "xslt")

# 디렉토리 존재 여부 확인 및 생성
for path in [XML_DIR, HTML_DIR]:
    os.makedirs(path, exist_ok=True)

# DB 연결
conn = pymysql.connect(host='localhost', user='root', password='', db='xml-sql', charset='utf8')
cursor = conn.cursor(pymysql.cursors.DictCursor)  # DictCursor로 설정

# 테이블 목록 얻기
def get_all_tables():
    cursor.execute("SHOW TABLES")
    key = list(cursor.description)[0][0]
    return [row[key] for row in cursor.fetchall()]

# 컬럼 존재 여부 체크 (DictCursor에 맞게 수정)
def has_column(table, column_name):
    cursor.execute(f"SHOW COLUMNS FROM `{table}`")
    return any(row["Field"] == column_name for row in cursor.fetchall())

# 고유값 가져오기
def get_distinct_values(table, column_name):
    cursor.execute(f"SELECT DISTINCT `{column_name}` FROM `{table}` WHERE `{column_name}` IS NOT NULL")
    return [row[column_name] for row in cursor.fetchall()]

# 조건에 맞는 행 가져오기
def get_filtered_rows(table, column_name, column_value):
    cursor.execute(f"SELECT * FROM `{table}` WHERE `{column_name}` = %s", (column_value,))
    return cursor.fetchall()
```

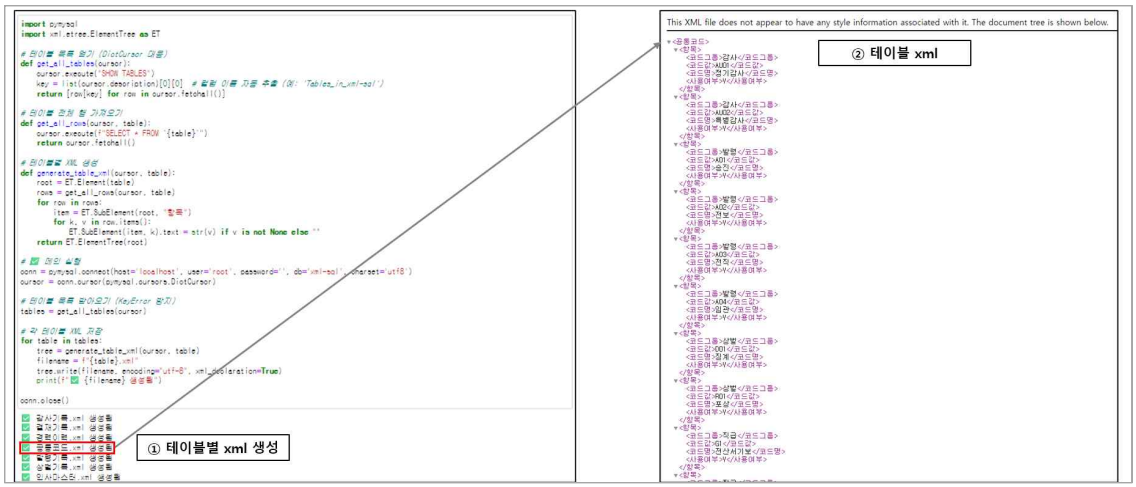
<그림 69> SQL-XML 추출 및 변환 코드

· Python 기반 SQL-XML 추출 및 변환 코드를 활용하여 조회한 SQL 테이블 구조 및 전체 데이터를 보존에 적합한 XML 형태로 변환함. 이 과정에서 데이터베이스의 테이블 구조를 XML 태그로 변환하고, 각 테이블 데이터를 XML 노드로 변환하는 작업이 수행됨

· 추출 대상은 검증 대상 정보시스템 데이터베이스의 모든 테이블을 포함함

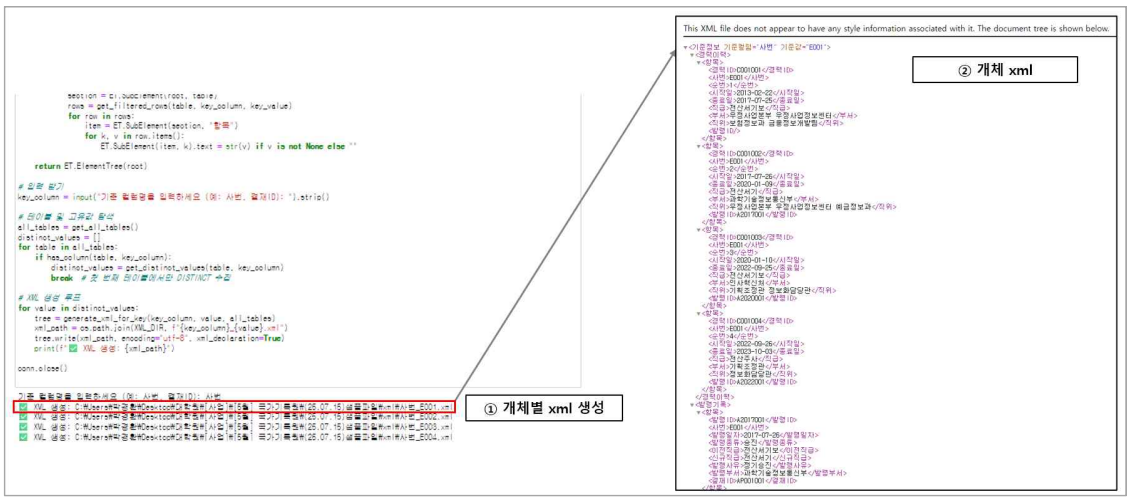
· 데이터 추출 방식은 (A) 테이블중심 XML 추출 방식(메인테이블.xml, 병역정보.xml 등)과 (B) 주요개체중심 XML 추출 방식(사번, 이름 등 개체별 통합 XML)으로 구분하여 적용함

· (A) 추출 방식을 통해 메인테이블.xml, 병역정보.xml 등과 같이 테이블을 중심으로 개별 XML 파일을 생성하여 테이블별 데이터를 독립적으로 관리할 수 있도록 구성함(<그림 70> 참조)



<그림 70> (A) 테이블중심 XML 추출 방식

· (B) 주요개체중심 XML 추출 방식을 통해 특정 객체나 컬럼을 중심으로 한 모든 정보를 통합한 XML 파일을 생성함. 이를 통해 한 개체의 전체 이력을 하나의 XML 파일에서 확인할 수 있도록 함(<그림 71> 참조)



<그림 71> (B) 주요개체중심 XML 추출 방식

- 재현 HTML 파일 생성 구조

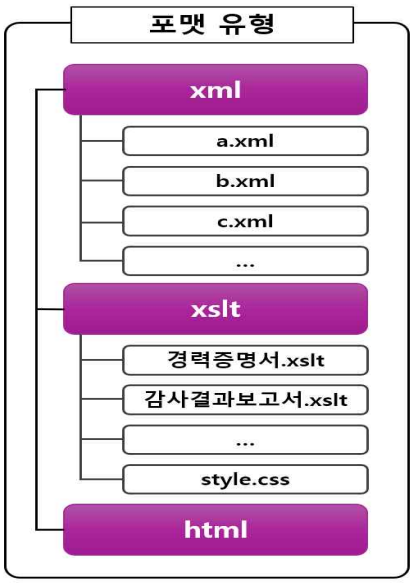
· 데이터(XML) 파일과 스타일(XSLT, CSS) 파일을 결합하여 검증 대상 정보시스템의 화면 레이아웃, 색상, 폰트, 표 구조 등을 재현한 HTML 파일들을 생성하고, 각 HTML 파일 간의 링크 연결이 가능하도록 구성함

· 데이터베이스로부터 추출 및 변환된 XML 데이터, XSLT, 재현용 HTML 파일을 <그림 72>와 같은 포맷 유형에 따라 각각 XML 폴더, XSLT 폴더, HTML 폴더에 저장함

  ◦ xml 폴더: 테이블/개체 단위로 추출된 a.xml, b.xml, c.xml 등의 XML 파일 저장

  ◦ xslt 폴더: 경력증명서.xslt, 감사결과보고서.xslt 등 문서 유형별 XSLT 파일 저장

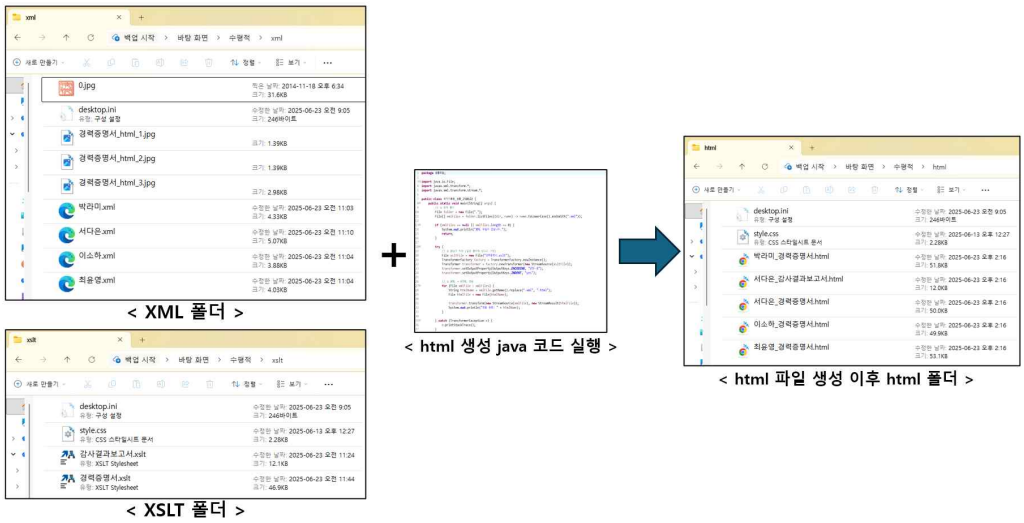
  ◦ html 폴더: 재현 결과 각 XML 파일에 대응되는 HTML 파일 저장



<그림 72> 재현 HTML 파일 생성 및 저장 구조

- 재현 HTML 파일 생성 과정

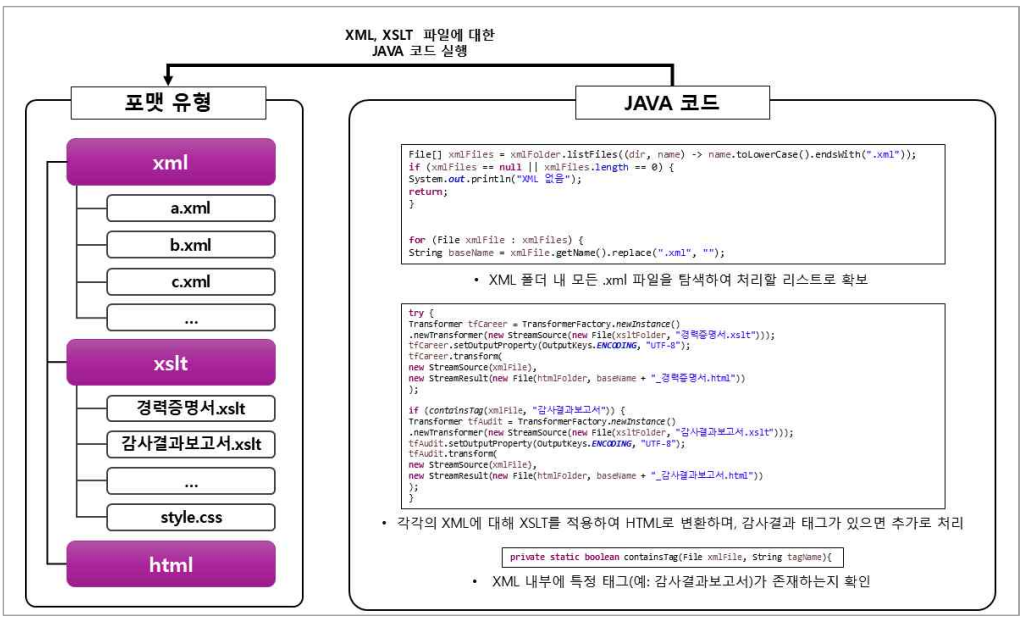
· XML 파일 내부의 구조를 분석하여 문서 유형별로 대응되는 XSLT 파일을 선택 적용하고 변환 결과물을 HTML 파일로 출력하기 위해 Java 기반 프로그래밍 코드를 활용함(<그림 73> 참조)



<그림 73> XML-XSLT 기반 html 파일 생성 과정

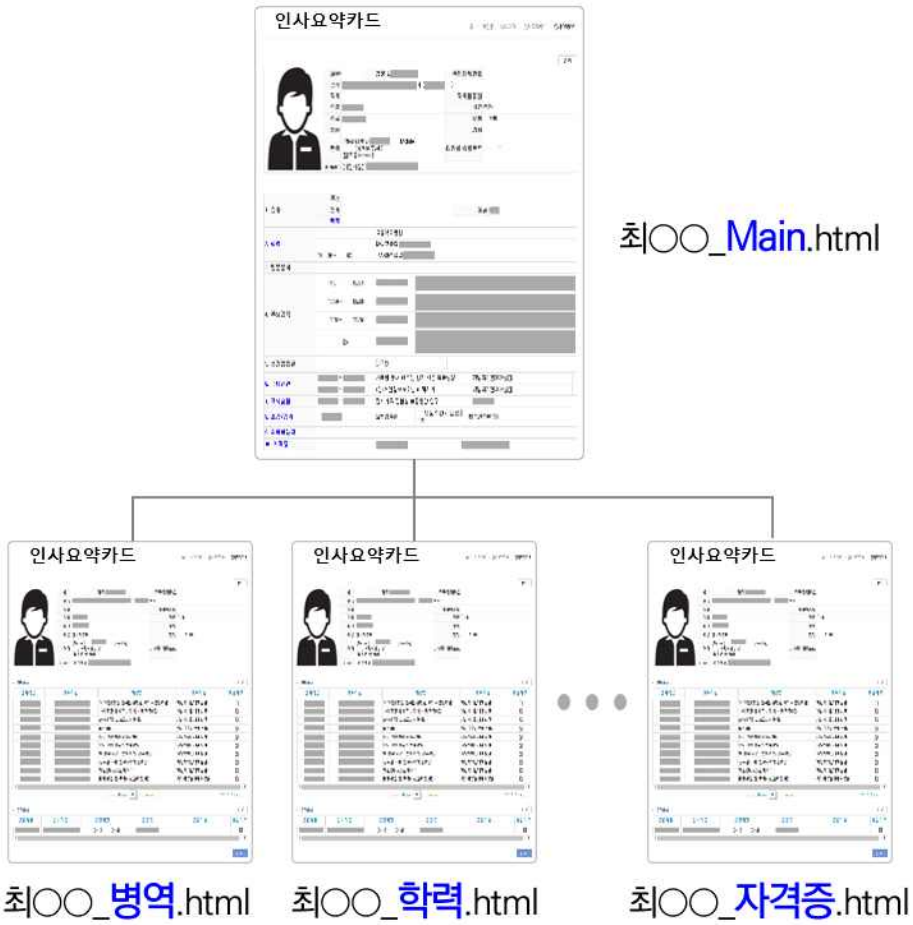
· Java 코드를 실행하여 xml 폴더 내의 모든 XML 파일을 탐색한 뒤, 해당 XML 파일의 문서 유형(예: 경력증명서, 감사결과보고서 등)을 판별함. 이후 xslt 폴더 내에서 XML 파일의 문서 유형과 일치하는 XSLT 파일을 XML 파일과 적용 및 결합하여 html 폴더 내에 HTML 파일을 생성함(<그림 74> 참조)

· 변환 프로그램은 특정 태그 존재 여부(예: <감사결과보고서></감사결과보고서>)를 확인하여 조건 분기 처리하도록 구성되어 있으며, 이 과정을 통해 동일한 XML 구조에서도 문서 유형별 HTML 파일을 자동 생성할 수 있음



<그림 74> Java 프로그래밍 기반 XML-XSLT 결합 HTML 파일 생성 과정

· 검증 대상 정보시스템 기반의 데이터세트 구조와 시각적 재현요소를 반영한 HTML 파일 생성을 위해 메인 HTML 파일은 인물별 기본정보와 주요 항목 목록을 상단에 요약 배치하고, 하단 탭을 통해 항목별 상세 페이지로 연결되는 구조로 구성함. 병역, 학력, 자격증 등 주요 정보 영역을 각각 분리한 상세 페이지는 개별 HTML 파일로 재현하며, 각 페이지는 해당 XML 데이터와 XSLT 변환 규칙을 기반으로 상호 연결됨(<그림 75> 참조)



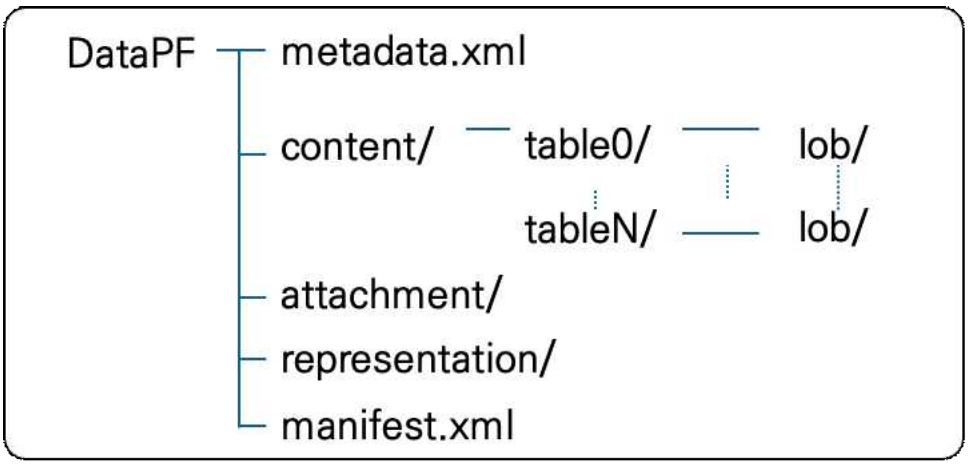
<그림 75> 상세 페이지 구성

나. 보존포맷 생성 및 검증

❍ DB형 데이터세트의 보존포맷 개요

- 전체 구조와 요소: metadata.xml(Trigger, Stored Procedure, View, Index 등)
- 실제 데이터: content/table*N* 폴더 만들어 table*N*.xml 파일에 저장
  · 폴더명/파일명: table*N*(*N*은 table 번호이며, 0부터 1씩 증가)
  · BLOB/CLOB 타입: 파일로 변환/생성하고, 각 tableN/lob 폴더에 저장
    ◦ 파일 이름: t*N* + *M* +원파일명(*N* 테이블 번호이고, *M*은 0부터 하나씩 증가)
  · 붙임 파일: attachment 폴더 아래에 붙임 파일 저장
    ◦ 파일이름 = t*N*_ + *M*′ + 원파일명(*N* 테이블 번호이고, *M*은 0부터 하나씩 증가)
    ◦ 예) Table0과 관련있는 첫번째 붙임 파일명: P001.pdf → t0_0_P001.pdf

· 재현 파일: representation 폴더 아래에 관련 파일을 저장

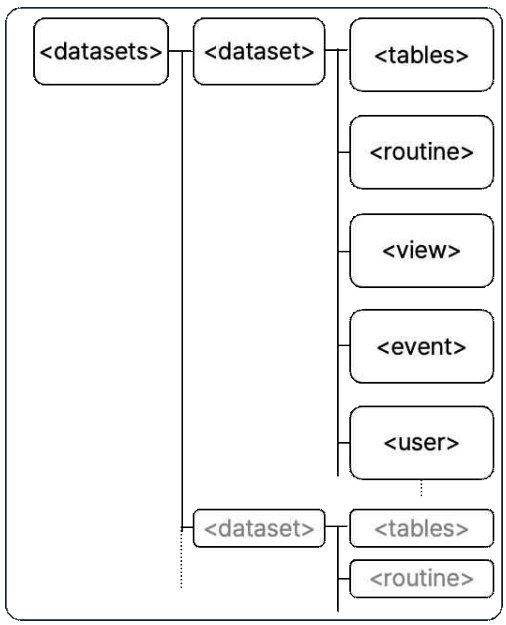


<그림 76> 보존 포맷 전체 폴더 구조

❍ metadata.xml 생성

- 개요

· metadata.xml은 데이터세트의 전체 구조와 요소를 정의한 xml 파일이며 보존 대상 DBMS의 table 정보, routine 정보, view 정보, event 정보, user 정보를 매칭되는 tag에 기술함

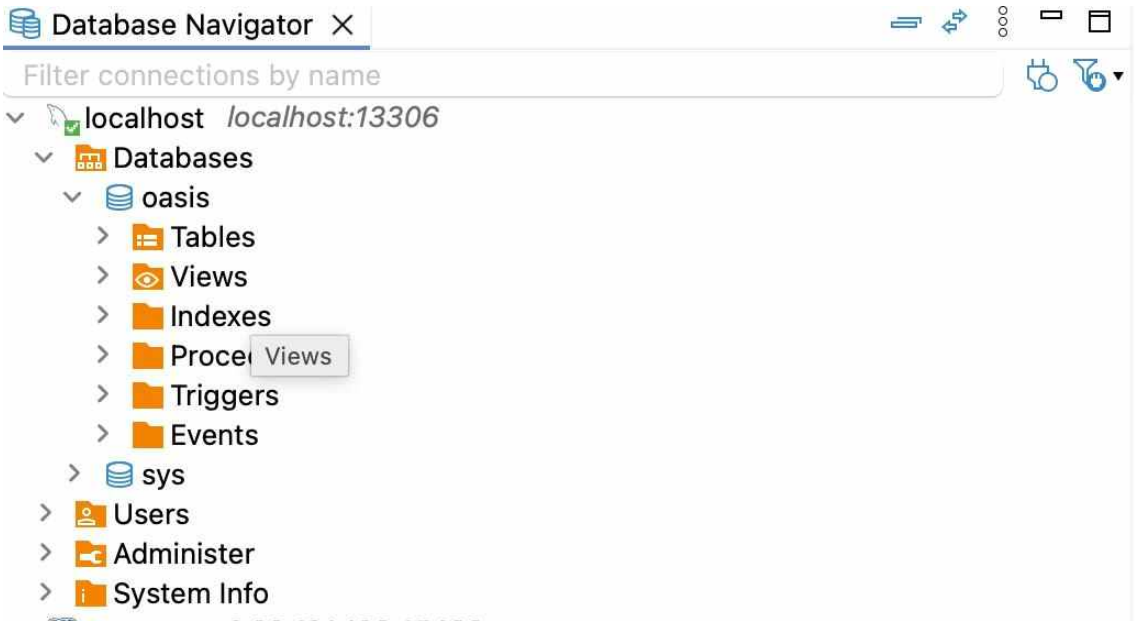


<그림 77> metadata.xml의 구조

- **tables 생성 과정**

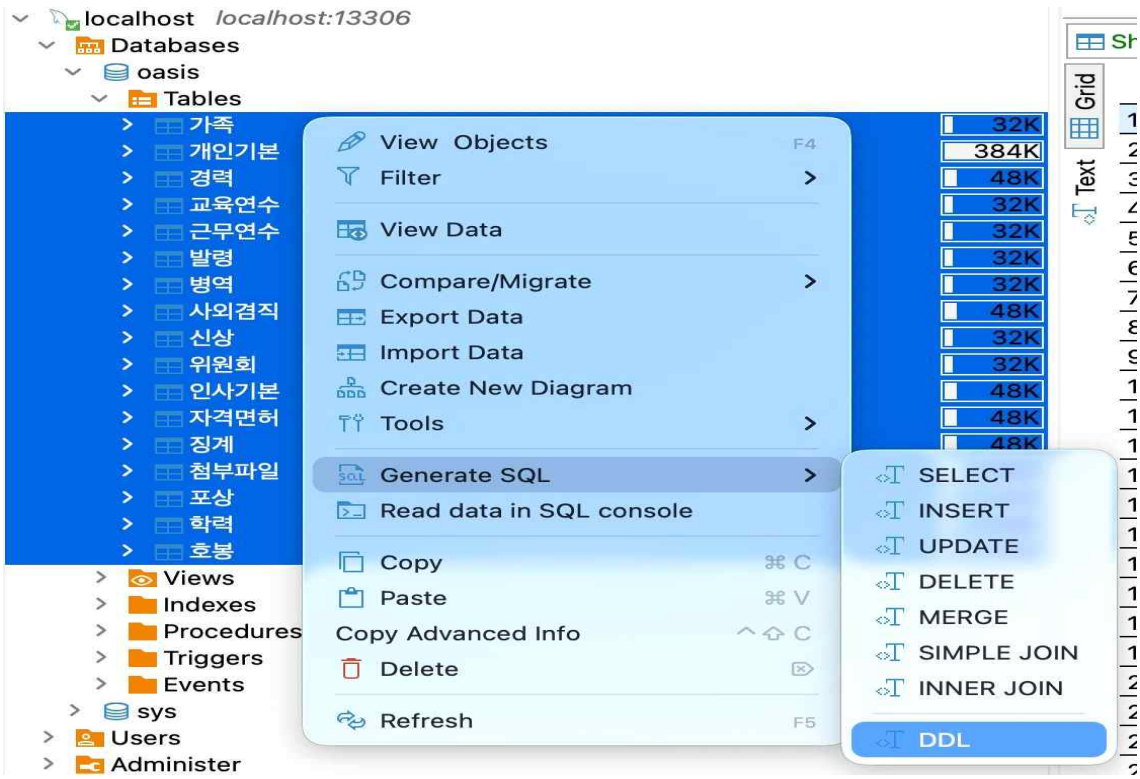
· (**오라클, Mysql, MSSQL**) 오픈 소스 DB Tool인 DBeaver(https://dbeaver.io/download/)를 활용하여 다음과 같이 DBMS상에 각 TABLE의 DDL문을 추출하여 metadata.xml의 tables tag 안에 기술

· DBeaver를 실행하고 보존 포맷 생성이 필요한 DB에 접속하면 다음과 같이 해당 DB의 모든 Object(Table, View, function, Procedure, Event 등)가 트리구조로 표시

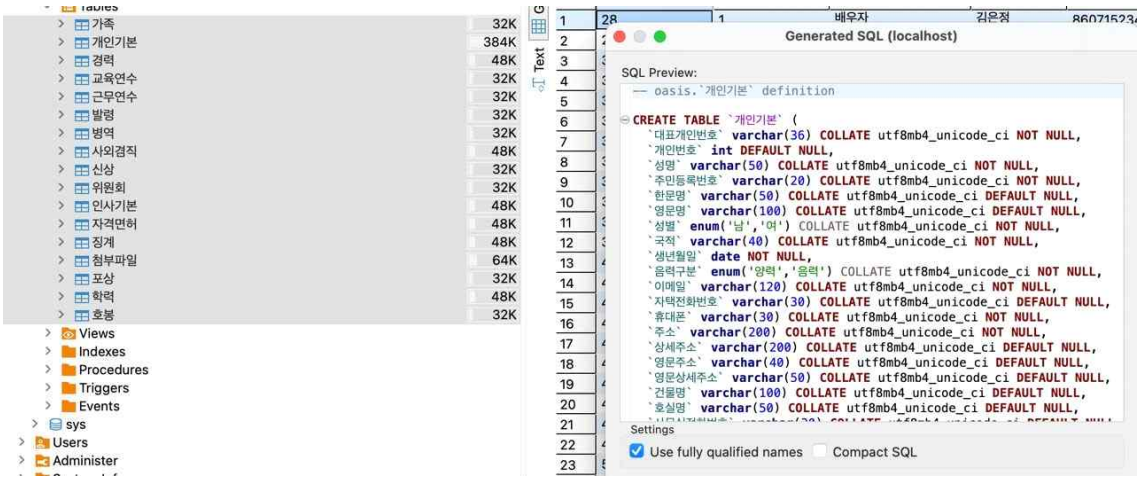


<그림 78> DBeaver 접속 화면

· 테이블에 대한 DDL 문을 저장할 경우 다음과 같이 Tables하위에 있는 모든 테이블을 선택한 후에 마우스 오른쪽 버튼을 클릭하고 컨텍스트 메뉴에서 Generate SQL > DDL을 선택하면 새로운 SQL 편집기 창에 전체 Table에 대한 DDL 스크립트가 표시(오라클/Mysql/MSSQL 동일)

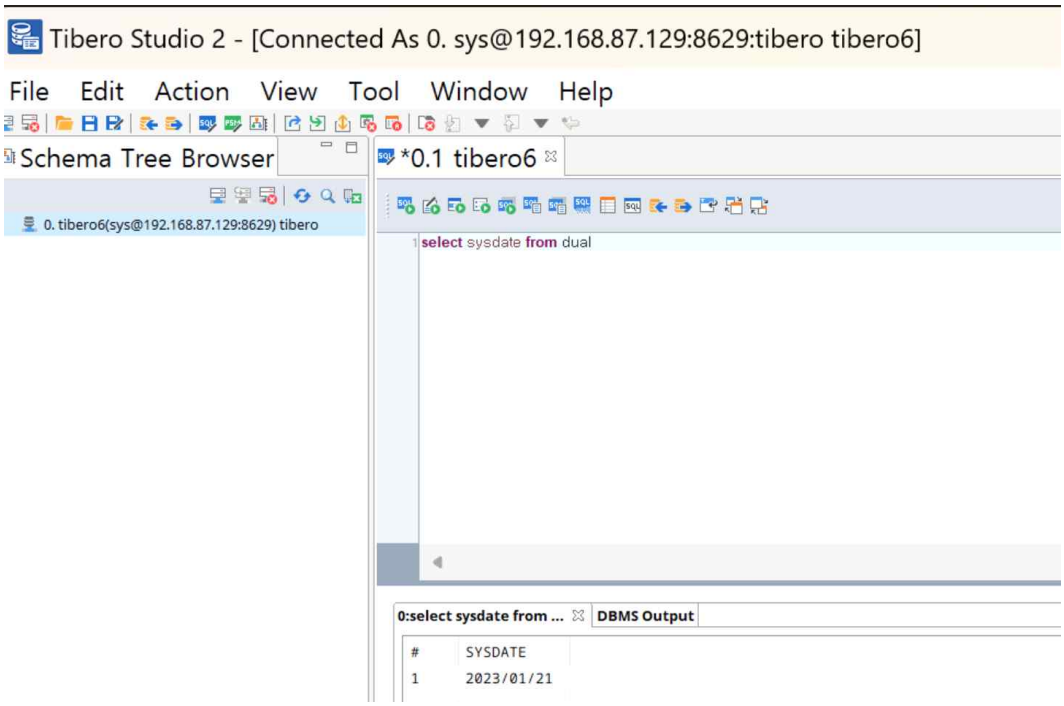


<그림 79> 전체 Table에 대한 DDL 추출 메뉴



<그림 80> 전체 Table에 대한 DDL 표시

· (**Tibero**) 티베로 공식 개발 도구인 Tibero Studio 실행한 후에 아래의 쿼리를 수행하면 모든 TABLE의 DDL문을 추출이 가능하고 metadata.xml의 tables tag 안에 각 테이블별로 DDL문을 복사하면 됨



<그림 81> Tibero Studio 실행 모습

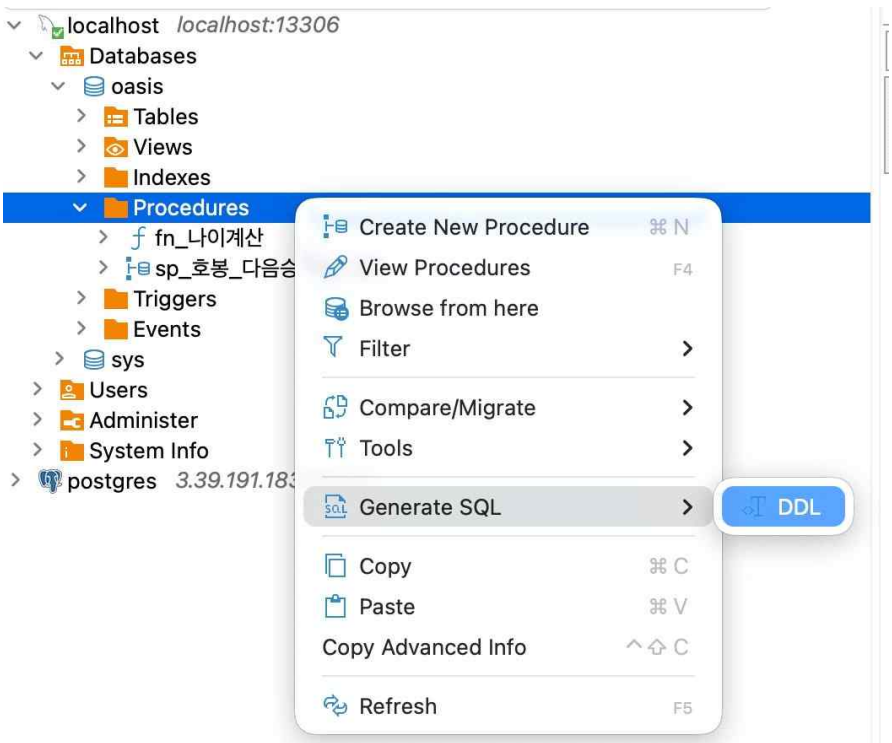
```
--table정보 추출 SQL문
SELECT DBMS_METADATA.GET_DDL('TABLE', table_name, owner) || ';' AS ddl_script
FROM ALL_TABLES
WHERE owner = 'YOUR_SCHEMA_NAME'; -- 'YOUR_SCHEMA_NAME'을 DB 스키마 이름으로 변경
```

· <datatasets><dataset><tables>....</tables></dataset></datatasets> 위치에 <그림 82>에서 추출된 table 정보를 metadata.xml의 tables tag에 CDATA를 이용하여 정보 삽입

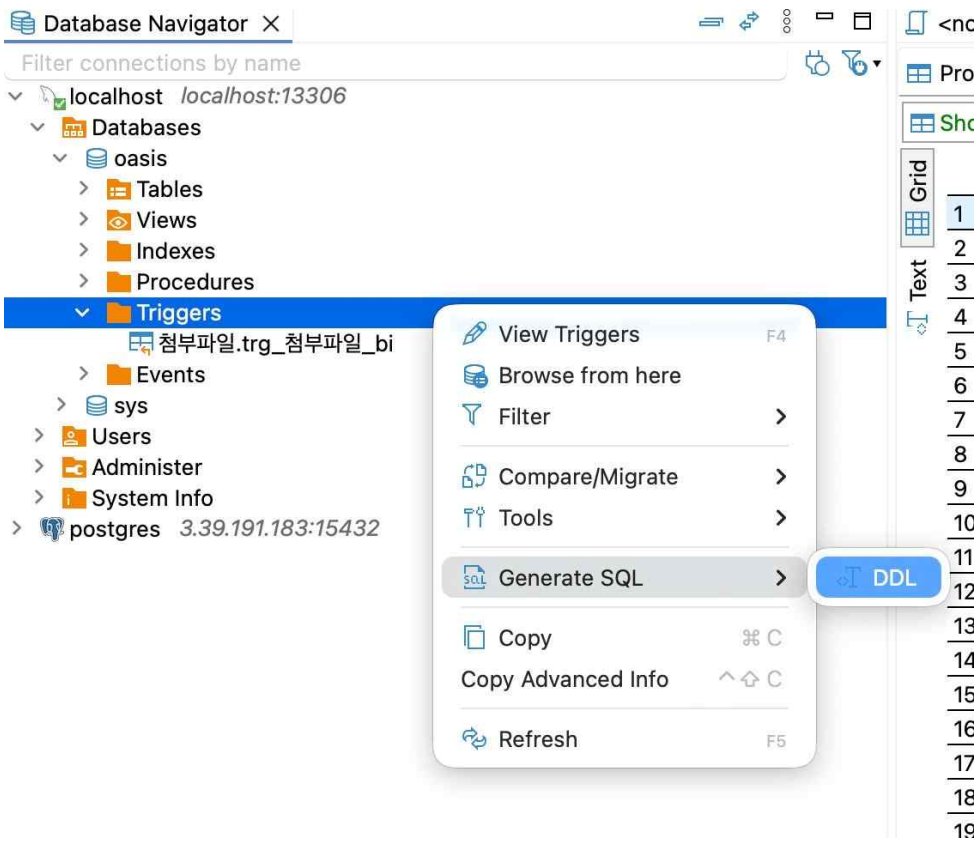
<그림 82> metadata.xml 내에 테이블 정보 예시

- routines 생성 과정

· (**오라클, Mysql, MSSQL**) Table과 동일하게 DBeaver 상에서 routine(Trigger, Procedure, Function) 정보 추출하여 metadata.xml의 routines tag 안에 기술



<그림 83> DBeaver를 이용한 Function/Procedure DDL 추출



<그림 84> DBeaver를 이용한 Trigger DDL 추출

```
--
-- Dumping routines for database 'oasis'
--
/*!50003 DROP FUNCTION IF EXISTS `fn_나이계산` */;
/*!50003 SET @saved_cs_client      = @@character_set_client */ ;
/*!50003 SET @saved_cs_results     = @@character_set_results */ ;
/*!50003 SET @saved_col_connection = @@collation_connection */ ;
/*!50003 SET character_set_client  = utf8mb4 */ ;
/*!50003 SET character_set_results = utf8mb4 */ ;
/*!50003 SET collation_connection  = utf8mb4_general_ci */ ;
/*!50003 SET @saved_sql_mode       = @@sql_mode */ ;
/*!50003 SET sql_mode              = 'NO_ZERO_IN_DATE,NO_ZERO_DATE,NO_ENGINE_SUBSTITUTION' */ ;
DELIMITER ;;
CREATE DEFINER=`root`@`localhost` FUNCTION `fn_나이계산`(생년월일 DATE) RETURNS int(11)
    DETERMINISTIC
BEGIN
  DECLARE 나이 INT;
  SET 나이 = YEAR(CURDATE()) - YEAR(생년월일);
  IF (DATE_FORMAT(CURDATE(), '%m%d') < DATE_FORMAT(생년월일, '%m%d')) THEN
    SET 나이 = 나이 - 1;
  END IF;
  RETURN 나이;
END ;;
```

<그림 85> DBeaver를 이용하여 추출된 function DDL 예시

· (**Tibero**) 티베로 공식 개발 도구인 Tibero Studio 실행한 후에 아래의 쿼리를 수행하여 모든 function/procedure/trigger의 DDL문을 추출이 가능

```
--function 추출 SQL문
SELECT    DBMS_METADATA.GET_DDL('FUNCTION',    object_name,    owner)    AS
ddl_script
FROM    ALL_OBJECTS
WHERE   object_type = 'FUNCTION'
AND     owner = 'YOUR_SCHEMA_NAME'; -- 'YOUR_SCHEMA_NAME'을 원하는 스
키마 이름으로 변경

--procedure 추출 SQL문
SELECT    DBMS_METADATA.GET_DDL('PROCEDURE',    object_name,    owner)    AS
ddl_script
FROM    ALL_OBJECTS
WHERE   object_type = 'PROCEDURE'
AND     owner = 'YOUR_SCHEMA_NAME'; -- 'YOUR_SCHEMA_NAME'을 원하는 스
키마 이름으로 변경

--trigger 추출 SQL문
SELECT    DBMS_METADATA.GET_DDL('TRIGGER',    trigger_name,    owner)    AS
ddl_script
FROM    ALL_TRIGGERS
WHERE   owner = 'YOUR_SCHEMA_NAME'; -- 'YOUR_SCHEMA_NAME'을 원하는 스
키마 이름으로 변경
```

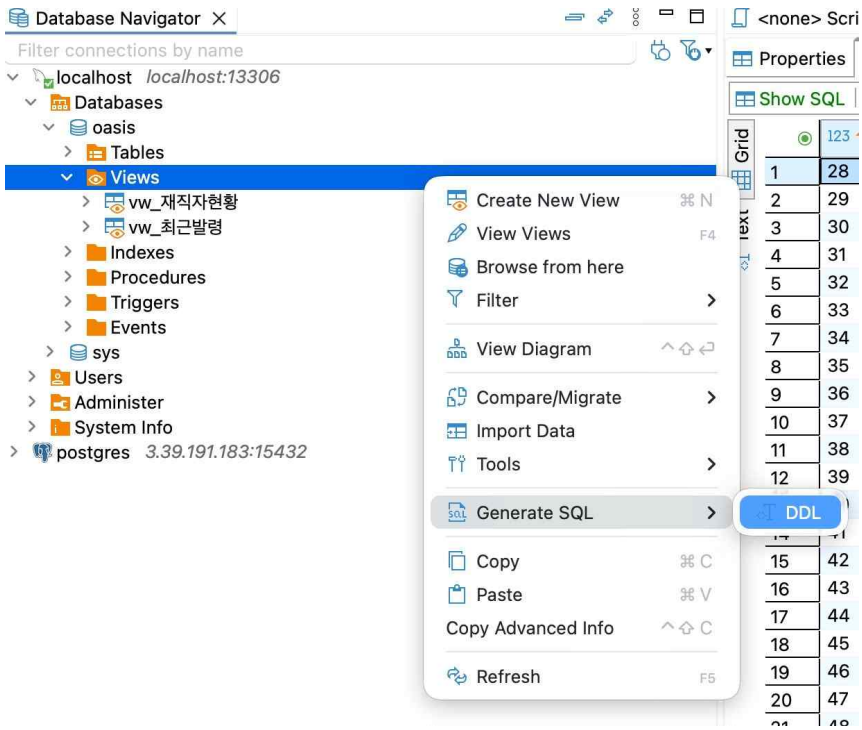
· <datatasets><dataset>**<routine>....</routine>**</dataset></datatasets> 위치에 <그림 86>에서 추출된 function정보를 metadata.xml의 routine tag에 CDATA를 이용하여 정보 삽입

```
<routine>
<![CDATA[
-- MySQL dump 10.13  Distrib 8.0.30, for Win64 (x86_64)
--
-- Host: localhost    Database: oasis
-- ------------------------------------------------------
-- Server version 8.0.30

--
-- Dumping routines for database 'oasis'
--
/*!50003 DROP FUNCTION IF EXISTS `fn_나이계산` */;
/*!50003 SET @saved_cs_client      = @@character_set_client */ ;
/*!50003 SET @saved_cs_results     = @@character_set_results */ ;
/*!50003 SET @saved_col_connection = @@collation_connection */ ;
/*!50003 SET character_set_client  = utf8mb4 */ ;
/*!50003 SET character_set_results = utf8mb4 */ ;
/*!50003 SET collation_connection  = utf8mb4_general_ci */ ;
/*!50003 SET @saved_sql_mode       = @@sql_mode */ ;
/*!50003 SET sql_mode              = 'NO_ZERO_IN_DATE,NO_ZERO_DATE,NO_ENGINE_SUBSTITUTION' */ ;
DELIMITER ;;
CREATE DEFINER=`root`@`localhost` FUNCTION `fn_나이계산`(생년월일 DATE) RETURNS int(11)
    DETERMINISTIC
BEGIN
  DECLARE 나이 INT;
  SET 나이 = YEAR(CURDATE()) - YEAR(생년월일);
  IF (DATE_FORMAT(CURDATE(), '%m%d') < DATE_FORMAT(생년월일, '%m%d')) THEN
    SET 나이 = 나이 - 1;
  END IF;
  RETURN 나이;
END ;;
```

<그림 86> metadata.xml의 routine 내에 function 정보 삽입

- **view 생성 과정**

· (**오라클, Mysql, MSSQL**) DBeaver 상에서 view 정보 추출하여 metadata.xml의 view tag 안에 기술



<그림 87> DBeaver를 이용한 View DDL 추출

```
--
-- Final view structure for view `vw_재직자현황`
--

/*!50001 DROP VIEW IF EXISTS `vw_재직자현황`*/;
/*!50001 SET @saved_cs_client          = @@character_set_client */;
/*!50001 SET @saved_cs_results         = @@character_set_results */;
/*!50001 SET @saved_col_connection     = @@collation_connection */;
/*!50001 SET character_set_client      = utf8mb4 */;
/*!50001 SET character_set_results     = utf8mb4 */;
/*!50001 SET collation_connection      = utf8mb4_general_ci */;
/*!50001 CREATE ALGORITHM=UNDEFINED */
/*!50013 DEFINER=`root`@`localhost` SQL SECURITY DEFINER */
/*!50001 VIEW `vw_재직자현황` AS select `i`.`인사기본ID` AS `인사기본ID`,`i`.`대표개인번호` AS `대표개인번호`,`i`.`근무부서` AS `근무부서`,`
/*!50001 SET character_set_client      = @saved_cs_client */;
/*!50001 SET character_set_results     = @saved_cs_results */;
/*!50001 SET collation_connection      = @saved_col_connection */;
```

<그림 88> 추출된 view(vw_재직자현황) DDL 예시

· (**Tibero**) 티베로 공식 개발 도구인 Tibero Studio 실행한 후에 아래의 쿼리를 수행하여 모든 View의 DDL문을 추출이 가능

```
SELECT DBMS_METADATA.GET_DDL('VIEW', view_name, owner) || ';' AS ddl_script
FROM ALL_VIEWS
WHERE owner = 'YOUR_SCHEMA_NAME'; -- 'YOUR_SCHEMA_NAME'을 원하는 스키마 이름으로 변경
```

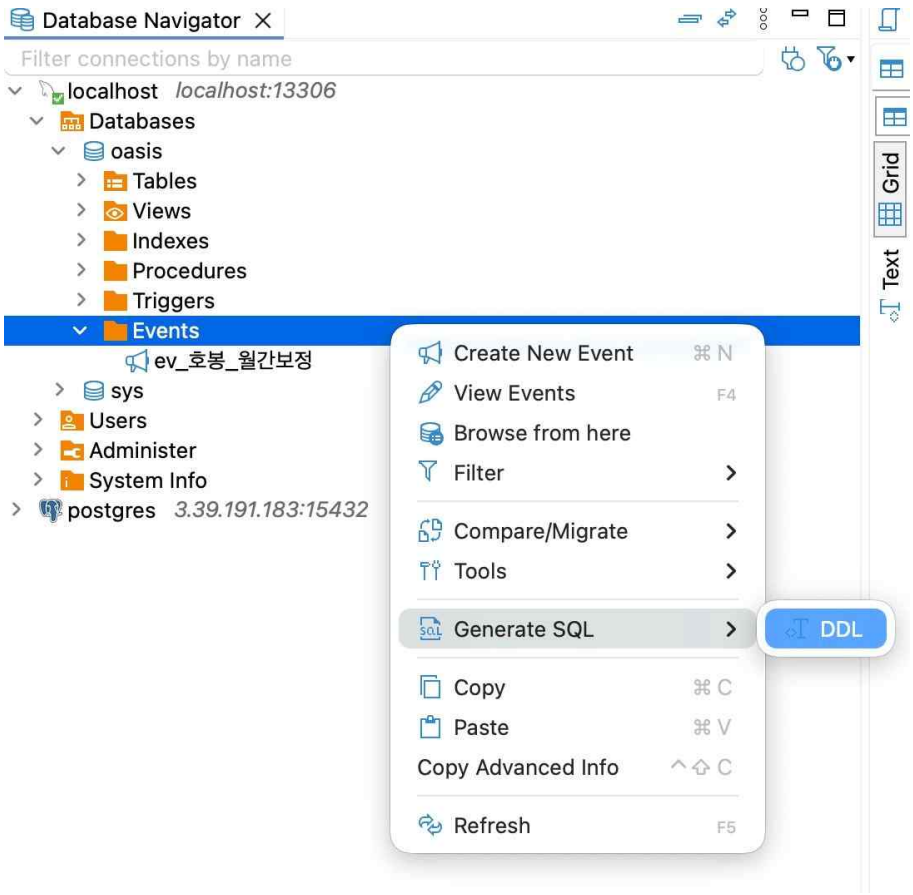
· <datatasets> <dataset><view>....</view></dataset></datatasets> 위치에 <그림 89>에서 추출된 view정보를 metadata.xml의 view tag에 CDATA를 이용하여 정보 삽입

```
<view>
    <![CDATA[
    -- MySQL dump 10.13  Distrib 8.0.30, for Win64 (x86_64)
--
-- Host: localhost    Database: oasis
-- ------------------------------------------------------
-- Server version 8.0.30

/*!40101 SET @OLD_CHARACTER_SET_CLIENT=@@CHARACTER_SET_CLIENT */;
/*!40101 SET @OLD_CHARACTER_SET_RESULTS=@@CHARACTER_SET_RESULTS */;
/*!40101 SET @OLD_COLLATION_CONNECTION=@@COLLATION_CONNECTION */;
/*!50503 SET NAMES utf8mb4 */;
/*!40103 SET @OLD_TIME_ZONE=@@TIME_ZONE */;
/*!40103 SET TIME_ZONE='+00:00' */;
/*!40014 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0 */;
/*!40014 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0 */;
/*!40101 SET @OLD_SQL_MODE=@@SQL_MODE, SQL_MODE='NO_AUTO_VALUE_ON_ZERO' */;
/*!40111 SET @OLD_SQL_NOTES=@@SQL_NOTES, SQL_NOTES=0 */;

--
-- Temporary view structure for view `vw_재직자현황`
--

DROP TABLE IF EXISTS `vw_재직자현황`;
/*!50001 DROP VIEW IF EXISTS `vw_재직자현황`*/;
SET @saved_cs_client     = @@character_set_client;
/*!50503 SET character_set_client = utf8mb4 */;
/*!50001 CREATE VIEW `vw_재직자현황` AS SELECT
 1 AS `인사기본ID`,
 1 AS `대표개인번호`,
 1 AS `근무부서`,
 1 AS `직급`,
 1 AS `재직상태구분`,
 1 AS `성명`,
 1 AS `이메일`*/;
SET character_set_client = @saved_cs_client;
```

<그림 89> metadata.xml의 view tag 내에 view DDL 정보 삽입

- **event 생성 과정**

· (**오라클, Mysql, MSSQL**) DBeaver 상에서 event 정보 추출하여 metadata.xml의 event tag 안에 기술



<그림 90> DBeaver를 이용한 Event DDL 추출

```
--
-- Dumping events for database 'oasis'
--
/*!50106 SET @save_time_zone= @@TIME_ZONE */ ;
/*!50106 DROP EVENT IF EXISTS `ev_호봉_월간보정` */;
DELIMITER ;;
/*!50003 SET @saved_cs_client      = @@character_set_client */ ;;
/*!50003 SET @saved_cs_results     = @@character_set_results */ ;;
/*!50003 SET @saved_col_connection = @@collation_connection */ ;;
/*!50003 SET character_set_client  = utf8mb4 */ ;;
/*!50003 SET character_set_results = utf8mb4 */ ;;
/*!50003 SET collation_connection  = utf8mb4_general_ci */ ;;
/*!50003 SET @saved_sql_mode       = @@sql_mode */ ;;
/*!50003 SET sql_mode              = 'NO_ZERO_IN_DATE,NO_ZERO_DATE,NO_ENGINE_SUBSTITUTION' */ ;;
/*!50003 SET @saved_time_zone      = @@time_zone */ ;;
/*!50003 SET time_zone             = 'SYSTEM' */ ;;
/*!50106 CREATE*/ /*!50117 DEFINER=`root`@`localhost`*/ /*!50106 EVENT `ev_호봉_월간보정` ON SCHEDULE EVERY 1 MONTH STARTS '2025-10-22 00:00:00' ON COMPLETION NOT PRESERVE ENABLE DO BEGIN
  CALL oasis.sp_호봉_다음승급일_계산();
END */ ;;
/*!50003 SET time_zone             = @saved_time_zone */ ;;
/*!50003 SET sql_mode              = @saved_sql_mode */ ;;
/*!50003 SET character_set_client  = @saved_cs_client */ ;;
/*!50003 SET character_set_results = @saved_cs_results */ ;;
/*!50003 SET collation_connection  = @saved_col_connection */ ;;
DELIMITER ;
/*!50106 SET TIME_ZONE= @save_time_zone */ ;
```

<그림 91> 추출된 event 예시

· (**Tibero**) 티베로 공식 개발 도구인 Tibero Studio 실행한 후에 아래의 쿼리를 수행하여 모든 컴포넌트(JOB, PROGRAM, SCHEDULE)의 DDL문을 추출이 가능(컴포넌트가 티베로에서는 Mysql의 Event에 해당함)

```
--Job 추출 SQL문
SELECT DBMS_METADATA.GET_DDL('JOB', job_name, owner) AS ddl_script
FROM   DBA_SCHEDULER_JOBS
WHERE  owner = 'YOUR_SCHEMA_NAME'; -- 특정 스키마명 입력

--Program 추출 SQL문
SELECT DBMS_METADATA.GET_DDL('PROGRAM', program_name, owner) AS
ddl_script
FROM   DBA_SCHEDULER_PROGRAMS
WHERE  owner = 'YOUR_SCHEMA_NAME';

--Schedule 추출 SQL문
SELECT DBMS_METADATA.GET_DDL('SCHEDULE', schedule_name, owner) AS
ddl_script
FROM   DBA_SCHEDULER_SCHEDULES
WHERE  owner = 'YOUR_SCHEMA_NAME';
```

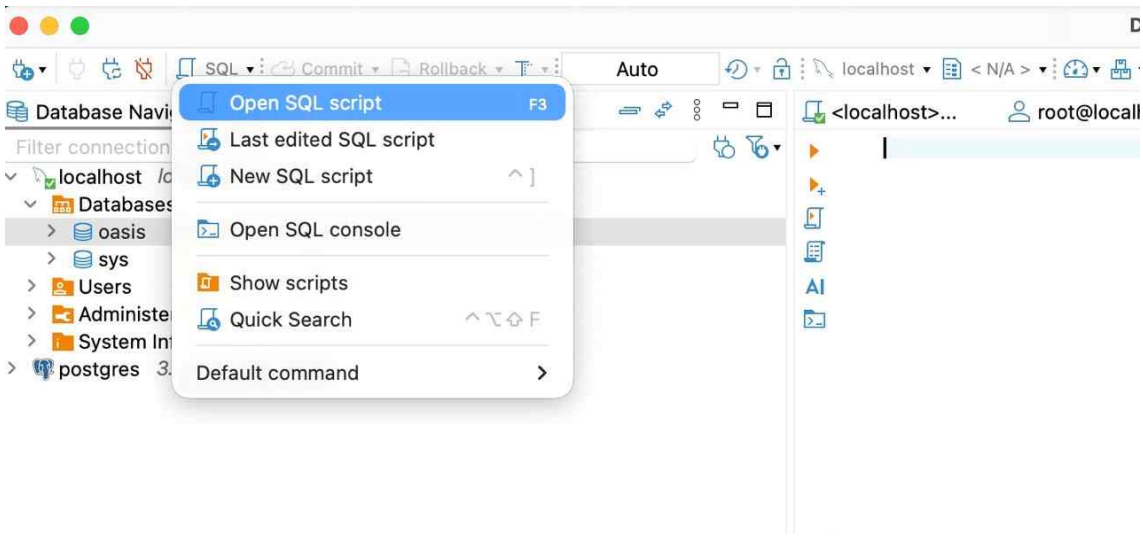
· <datatasets><dataset><event>....</event></dataset></datatasets> 위치에 <그림 92>에서 추출된 event 정보를 metadata.xml의 event tag에 CDATA를 이용하여 정보 삽입

```
<event>
    <![CDATA[
--
-- Dumping events for database 'oasis'
--
/*!50106 SET @save_time_zone= @@TIME_ZONE */ ;
/*!50106 DROP EVENT IF EXISTS `ev_호봉_월간보정` */;
DELIMITER ;;
/*!50003 SET @saved_cs_client      = @@character_set_client */ ;;
/*!50003 SET @saved_cs_results     = @@character_set_results */ ;;
/*!50003 SET @saved_col_connection = @@collation_connection */ ;;
/*!50003 SET character_set_client  = utf8mb4 */ ;;
/*!50003 SET character_set_results = utf8mb4 */ ;;
/*!50003 SET collation_connection  = utf8mb4_general_ci */ ;;
/*!50003 SET @saved_sql_mode       = @@sql_mode */ ;;
/*!50003 SET sql_mode              = 'NO_ZERO_IN_DATE,NO_ZERO_DATE,NO_ENGINE_SUBSTITUTION' */ ;;
/*!50003 SET @saved_time_zone      = @@time_zone */ ;;
/*!50003 SET time_zone             = 'SYSTEM' */ ;;
/*!50106 CREATE*/ /*!50117 DEFINER=`root`@`localhost`*/ /*!50106 EVENT `ev_호봉_월간보정` ON SCHEDULE EVERY 1
  CALL oasis.sp_호봉_다음승급일_계산();
END */ ;;
```

<그림 92> metadata.xml의 event tag내에 event DDL 정보 삽입

- **user/grant 생성 과정**

· **(오라클/Mysql/Sqlserver)** DBeaver를 이용하여 DBeaver 상단 메뉴에서 SQL 편집기 열기 버튼을 클릭하거나 F3 키를 누른 후에 아래의 쿼리를 실행하여 추출



<그림 93> DBeaver에서 SQL 편집기 실행

· (**오라클**) user 및 권한 추출 SQL

```
-- 특정 사용자 이름에 대한 DDL 추출 (여기서 'YOUR_USERNAME'을 실제 사용자 이름으로 변경)
SELECT DBMS_METADATA.GET_DDL('USER', 'YOUR_USERNAME') AS DDL_Source FROM DUAL;

-- 해당 사용자에게 부여된 시스템 권한(예: CONNECT, RESOURCE) 추출
SELECT DBMS_METADATA.GET_GRANTED_DDL('SYSTEM_GRANT', 'YOUR_USERNAME') AS DDL_Source FROM DUAL;

-- 해당 사용자에게 부여된 역할(Role) 추출
SELECT DBMS_METADATA.GET_GRANTED_DDL('ROLE_GRANT', 'YOUR_USERNAME') AS DDL_Source FROM DUAL;

-- 해당 사용자에게 부여된 테이블/객체 권한 추출 (필요 시 주석 해제하여 사용)
SELECT DBMS_METADATA.GET_GRANTED_DDL('OBJECT_GRANT', 'YOUR_USERNAME') AS DDL_Source FROM DUAL;
```

· (**Mysql**) user 및 권한 추출 SQL

```
-- create user ddl 추출
SELECT CONCAT('CREATE USER '', user, ''@'', host, '' IDENTIFIED BY PASSWORD '', authentication_string, '';') AS create_user_ddl
FROM mysql.user
WHERE host NOT IN ('localhost', '::1') AND user NOT IN ('mysql.infoschema', 'mysql.session', 'mysql.sys', 'root');
-- 필요에 따라 특정 사용자나 호스트를 필터링할 수 있습니다.

--권한 부여(grant) ddl 추출
SHOW GRANTS FOR 'username'@'host';
```

· (**Sqlserver**) user 및 권한 추출 SQL

```
-- 데이터베이스 사용자의 역할(Role) 확인
-- 'YourDatabase'를 실제 데이터베이스 이름으로 변경
USE YourDatabase;
GO

SELECT
    dp.name AS RoleName,
    mp.name AS MemberName
FROM
    sys.database_role_members drm
INNER JOIN
    sys.database_principals dp ON drm.role_principal_id = dp.principal_id
INNER JOIN
    sys.database_principals mp ON drm.member_principal_id = mp.principal_id
WHERE
    mp.name = 'YOUR_USERNAME'; -- 'YOUR_USERNAME'을 실제 사용자 이름으로 변경

-- 특정 사용자에게 부여된 객체 권한 확인 (Object Grants)
-- 'YourDatabase'를 실제 데이터베이스 이름으로 변경
USE YourDatabase;
GO

SELECT
    state_desc + ' ' + permission_name + ' ON ' + OBJECT_NAME(major_id) + ' TO ' + USER_NAME(grantee_principal_id) AS GrantStatement
FROM
    sys.database_permissions
WHERE
    class = 1 -- Object or column
    AND grantee_principal_id = USER_ID('YOUR_USERNAME'); -- 'YOUR_USERNAME'을 실제 사용자 이름으로 변경
```

· (**Tibero**) Tibero Studio 실행 후 아래의 쿼리를 수행하여 user 및 권한 정보 추출

```
--user 추출 SQL문
SELECT TO_CHAR(DBMS_METADATA.GET_DDL('USER', 'TIBERO')) FROM DUAL;

--권한 추출 SQL문
SELECT TO_CHAR(DBMS_METADATA.GET_GRANTED_SQL('USER_GRANT', 'TIBERO')) FROM DUAL;
```

```
-- ====================================
-- 사용자 계정 및 역할/권한 정의 (추가)
-- ====================================

-- 사용자 생성
CREATE USER `업무담당자A`@'%' IDENTIFIED BY 'passA!';
CREATE USER `업무담당자B`@'%' IDENTIFIED BY 'passB!';
CREATE USER `시스템담당자C`@'%' IDENTIFIED BY 'passC!';
CREATE USER `시스템담당자D`@'%' IDENTIFIED BY 'passD!';
CREATE USER `기록연구사`@'%' IDENTIFIED BY 'archivist!';

-- 권한 부여
GRANT SELECT, INSERT, UPDATE, DELETE ON oasis.* TO `업무담당자A`@'%', `업무담당자B`@'%';
GRANT ALL PRIVILEGES ON oasis.* TO `시스템담당자C`@'%', `시스템담당자D`@'%';
GRANT SELECT ON oasis.* TO `기록연구사`@'%';

FLUSH PRIVILEGES;
```

<그림 94> 추출된 user 생성문 예시

· <datatasets> <dataset><user>....</user></dataset></datatasets> 위치에 <그림 95>에서 추출된 view정보를 metadata.xml의 view tag에 CDATA를 이용하여 정보 삽입

```
<user>
<![CDATA[
-- ====================================
-- 사용자 계정 및 역할/권한 정의 (추가)
-- ====================================

-- 사용자 생성
CREATE USER `업무담당자A`@'%' IDENTIFIED BY 'passA!';
CREATE USER `업무담당자B`@'%' IDENTIFIED BY 'passB!';
CREATE USER `시스템담당자C`@'%' IDENTIFIED BY 'passC!';
CREATE USER `시스템담당자D`@'%' IDENTIFIED BY 'passD!';
CREATE USER `기록연구사`@'%' IDENTIFIED BY 'archivist!';

-- 권한 부여
GRANT SELECT, INSERT, UPDATE, DELETE ON oasis.* TO `업무담당자A`@'%', `업무담당자B`@'%';
GRANT ALL PRIVILEGES ON oasis.* TO `시스템담당자C`@'%', `시스템담당자D`@'%';
GRANT SELECT ON oasis.* TO `기록연구사`@'%';

FLUSH PRIVILEGES;
```

<그림 95> metadata.xml의 user tag내에 user 생성 sql 삽입

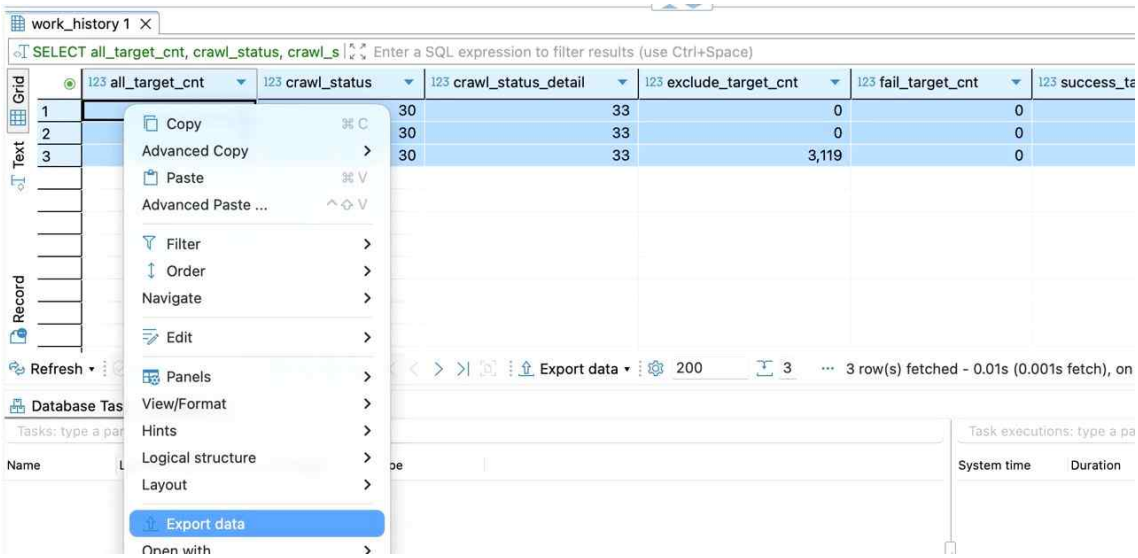
❍ table별 Content 생성 및 XML 변환

- 개요

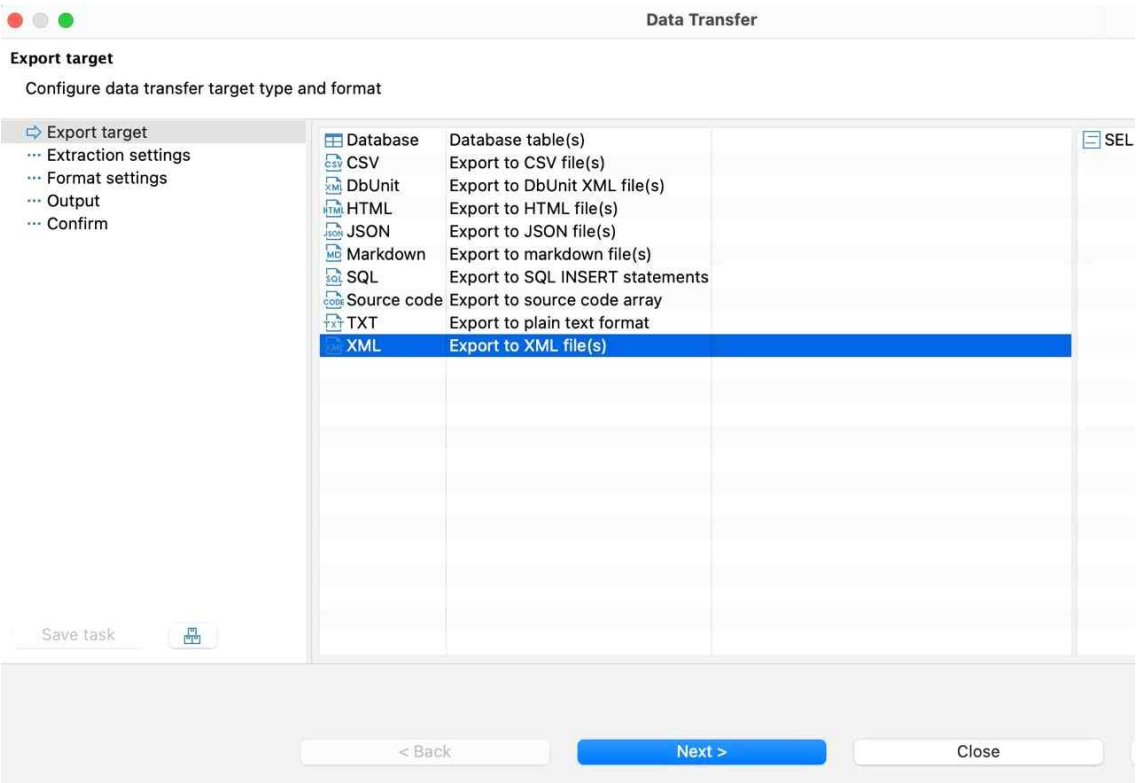
· DBeaver를 이용하여 DBeaver 상단 메뉴에서 SQL 편집기 열기 버튼을 클릭하거나 F3 키를 누른 후에 편집기 실행

· 데이터세트를 조회할 SELECT 문을 작성하여 SQL Editor에서 실행

· 하단의 “Grid”에서 SQL 실행 결과를 검토하고 전체 데이터를 선택한 후에 마우스 오른쪽 버튼을 클릭하여 Context 메뉴에서 “Export data” 메뉴 선택



· 다음과 같이 옵션에서 XML 선택



· 파일 이름과 저장 위치를 지정하여 XML로 저장

· XML로 Export된 결과와 Query수행 결과를 비교하여 정합성 확인

```xml
<?xml version="1.0" encoding="UTF-8"?>
<work_history>
  <DATA_RECORD>
    <all_target_cnt>0</all_target_cnt>
    <crawl_status>30</crawl_status>
    <crawl_status_detail>33</crawl_status_detail>
    <exclude_target_cnt>0</exclude_target_cnt>
    <fail_target_cnt>0</fail_target_cnt>
    <success_target_cnt>0</success_target_cnt>
    <target_size>4,380</target_size>
    <target_work_cnt>0</target_work_cnt>
    <crawl_end_at>2025-07-07 05:57:13.858 +0900</crawl_end_at>
    <crawl_start_at>2025-07-07 05:55:13.858 +0900</crawl_start_at>
    <mod_at>2025-07-07 05:57:13.858 +0900</mod_at>
    <reg_at>2025-09-05 16:10:00.929 +0900</reg_at>
    <target_site_id>1</target_site_id>
    <work_history_id>1f08a275-6904-6dd4-b407-c1b98fac0459</work_history_id>
    <work_server_id>2</work_server_id>
    <mod_username>비회원</mod_username>
    <reg_username>admin</reg_username>
    <re_site_time>20250707055513</re_site_time>
    <target_result_ext>warc</target_result_ext>
    <host_name>https://www.kasa.go.kr</host_name>
    <active>1</active>
    <publish>1</publish>
  </DATA_RECORD>
  <DATA_RECORD>
    <all_target_cnt>0</all_target_cnt>
    <crawl_status>30</crawl_status>
    <crawl_status_detail>33</crawl_status_detail>
    <exclude_target_cnt>0</exclude_target_cnt>
    <fail_target_cnt>0</fail_target_cnt>
    <success_target_cnt>0</success_target_cnt>
    <target_size>213</target_size>
    <target_work_cnt>0</target_work_cnt>
    <crawl_end_at>2025-07-31 08:10:32.105 +0900</crawl_end_at>
    <crawl_start_at>2025-07-31 08:08:32.105 +0900</crawl_start_at>
```

· blob 타입 컬럼이 포함된 경우 다음과 같이 각 DBMS의 base64 인코딩 함수를 이용하여 binary data를 문자열로 converting 하여 xml내에 문자열로 저장

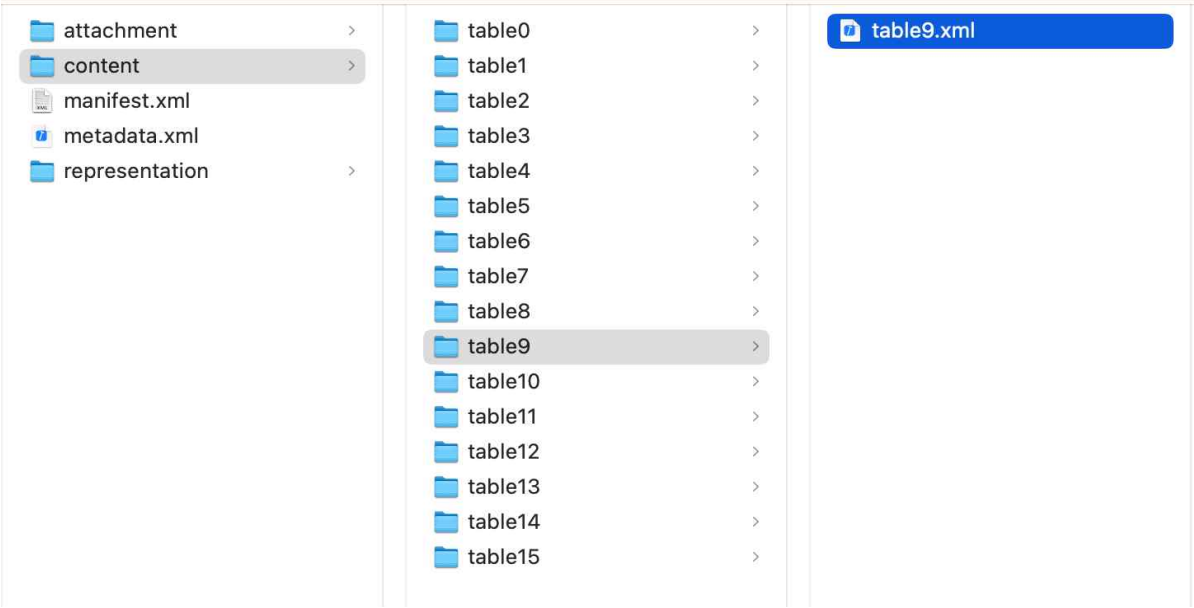
```
SELECT `건물명`, `국적`, `대표개인번호`, `대표보직사무실전화번호`, `사무실전화번호`, `상세주소`, `생년월일`, `생성일시`, `성명`, `성별`, `수정일시`, `영문명`, `음력구분`, `이메일`, `자택전화번호`, `주민등록번호`, `주소`, `팩스`, `한문명`, `호실명`, `휴대폰`, TO_BASE64(`프로필사진`) FROM `oasis`.`개인기본`;
```

<그림 96> blob 컬럼의 데이터를 base64로 인코딩하여 XML로 추출한 예시

· 위에서 저장된 XML 파일을 metadata.xml에 기술된 내용에 맞게 table*N*.xml(*N*은 table 번호이며, 0부터 1씩 증가)로 파일 이름을 변경하여 metadata.xml에 명시된 폴더(content/table*N*)에 위치

```
<table datapath="file:///content/table0/table9.xml" name="인사기본">
<![CDATA[
--
-- Table structure for table `인사기본`
--

DROP TABLE IF EXISTS `인사기본`;
/*!40101 SET @saved_cs_client     = @@character_set_client */;
/*!50503 SET character_set_client = utf8mb4 */;
CREATE TABLE `인사기본` (
  `인사기본ID` bigint NOT NULL,
  `대표개인번호` varchar(36) COLLATE utf8mb4_unicode_ci NOT NULL,
  `소속기관` varchar(120) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `근무부서` varchar(120) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `재직상태구분` enum('재직','휴직','파견','겸직','퇴직','정년퇴직') COLLATE utf8mb4_unicode_ci DEFAULT '재직',
  `개인번호` varchar(20) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `신분구분` varchar(40) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `현직종` varchar(60) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `직급` varchar(40) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `근속가봉` int DEFAULT NULL,
  `대우필수` varchar(20) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `급여지급기관` varchar(120) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `관리대학` varchar(100) COLLATE utf8mb4_unicode_ci DEFAULT NULL,
  `생성일시` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,
  `수정일시` timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON UPDATE CURRENT_TIMESTAMP,
  PRIMARY KEY (`인사기본ID`),
  UNIQUE KEY `uk_person_main` (`대표개인번호`),
```

<그림 97> metadata.xml 내에 해당 테이블 데이터 파일 및 DDL 정보 예시



<그림 98> metadata.xml에 명시된 위치에 tableN.xml 저장

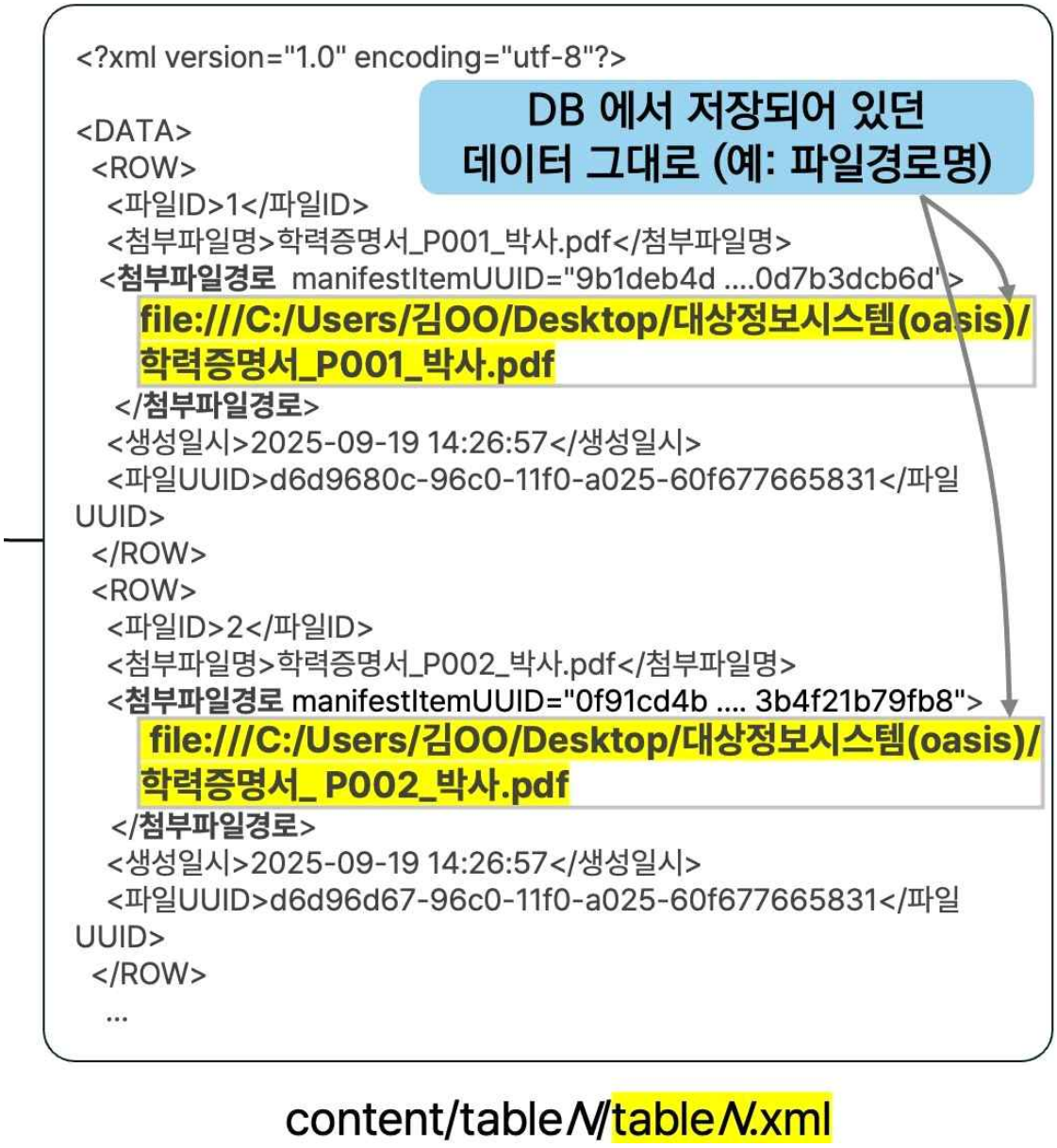
○ 첨부파일 정보 저장(attachment/ 폴더 생성)

- 개요

· 테이블의 첨부파일 컬럼에 저장된 실제 첨부파일을 attachment/ 하위에 저장하고 해당 위치를 manifext.xml에 기술
(manifestItemUUID를 통해서 table*N*.xml내의 원본 파일 경로와 맵핑됨)

- XML 생성 프로세스

· 첨부파일이 포함된 테이블의 데이터를 조회하는 query를 통해서 필요한 데이터를 조회하고 XML로 저장



<그림 99> 원본 첨부파일 경로가 포함된 tableN.xml 예시

- 원본 첨부파일 경로는 table*N*.xml내의 첨부파일경로에 저장하고 attachment/ 폴더에 저장된 실제 파일 경로는 manifext.xml에 기술(manifestItemUUID로 맵핑)

<그림 100> manifext.xml내에 manifestItemUUID를 기준으로 보존포맷 내의 실제 첨부파일 경로 표기

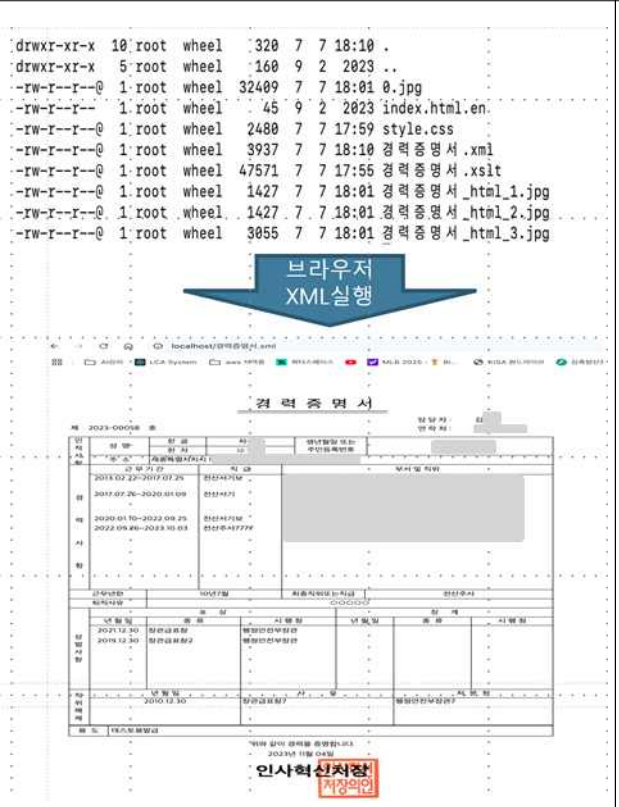
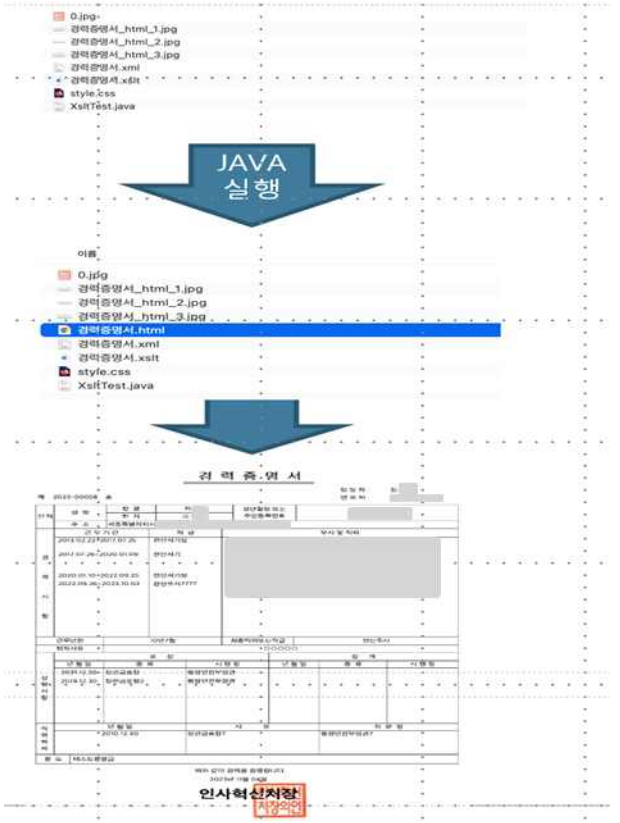
다. 보존포맷 재현도구(XSLT/XML) 생성 및 웹기반 기록물 재현

❍ 웹기록물 재현 개요

- 보존포맷의 재현 프로세스는 개체 중심 XML 생성, XSLT 시각화, HTML 생성(표시), 요소 단위 링크 구성의 단계적 흐름으로 체계화. 화면의 HTML 소스 코드 분석을 통해 테이블 구조와 CSS 패턴을 파악하고, XML 데이터를 동일한 시각적 결과로 변환하는 XSLT 추출 방안을 검토. 아래 <표 98>과 같이 HTML 형태의 원본 웹페이지를 브라우저를 이용해서 재현하는 방법은 크게 웹서버 방식과 프로그램 방식으로 나뉘어짐

- 웹서버 방식은 웹서버가 설치된 환경에서 별도의 프로그램없이 XML파일을 브라우저에서 실행할 경우 재현 룰을 정의한 XSLT와 결합되어 원본 화면이 재현되며, 프로그램 방식의 경우 XML과 XSLT파일이 프로그램 명령어의 의해서 HTML이 생성되고 해당 HTML을 브라우저에서 실행하면 원본 화면이 재현되는 것을 확인할 수 있음

<표 98> [재현 방식] 웹서버 방식 VS 프로그램 방식

웹서버 방식	프로그램 방식
	
· 웹서버가 설치된 환경에서 xml파일을 브라우저에서 실행시키면 원본 웹페이지를 확인할 수 있음	· XML과 XSLT가 있는 폴더에서 프로그램 명령어를 실행시키면 HTML이 생성되며 해당 HTML은 브라우저를 통해 확인 가능

○ 웹기록물 재현도구(XSLT/XML) 생성 프로세스

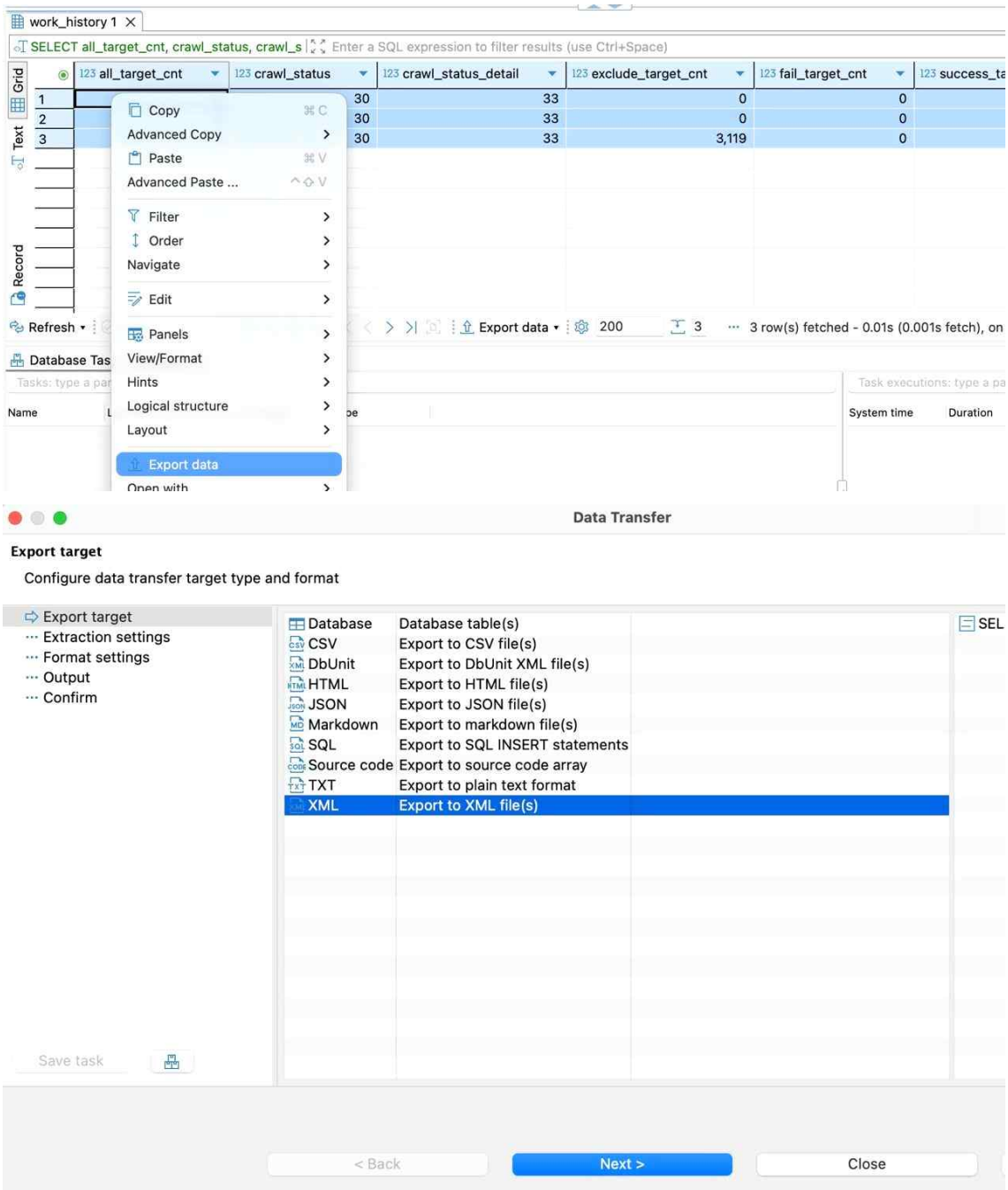
- 웹기록물 재현하기 위해서는 <표 99>과 같이 3개의 구성요소가 필요하며 실제 재현 시점에는 XML과 XSLT를 이용하여 원본 화면과 동일하게 재현하는 것이 가능함

<표 99> 재현도구 구성 요소

요소	내용
XML	· 재현이 필요한 화면의 데이터 XML(DB Query를 통해서 추출)
HTML/CSS	· 재현이 필요한 화면의 HTML 소스 코드(보통 HTML과 HTML의 style을 정의하는 css로 구성이 되어 있음) · css는 별도 파일로 추출하여 HTML상에서 css 파일을 import하여 원본 화면을 재현하는데 사용됨
XSLT	· HTML을 XML형태로 변환한 XML기반의 Stylesheet문서

- XML 생성

· 쿼리를 통해서 재현이 필요한 화면의 데이터를 XML로 추출



<그림 104> 오픈소스 Tool을 이용하여 Query 수행 결과를 XML로 저장

```xml
<?xml version="1.0" encoding="UTF-8"?>
<work_history>
  <DATA_RECORD>
    <all_target_cnt>0</all_target_cnt>
    <crawl_status>30</crawl_status>
    <crawl_status_detail>33</crawl_status_detail>
    <exclude_target_cnt>0</exclude_target_cnt>
    <fail_target_cnt>0</fail_target_cnt>
    <success_target_cnt>0</success_target_cnt>
    <target_size>4,380</target_size>
    <target_work_cnt>0</target_work_cnt>
    <crawl_end_at>2025-07-07 05:57:13.858 +0900</crawl_end_at>
    <crawl_start_at>2025-07-07 05:55:13.858 +0900</crawl_start_at>
    <mod_at>2025-07-07 05:57:13.858 +0900</mod_at>
    <reg_at>2025-09-05 16:10:00.929 +0900</reg_at>
    <target_site_id>1</target_site_id>
    <work_history_id>1f08a275-6904-6dd4-b407-c1b98fac0459</work_history_id>
    <work_server_id>2</work_server_id>
    <mod_username>비회원</mod_username>
    <reg_username>admin</reg_username>
    <re_site_time>20250707055513</re_site_time>
    <target_result_ext>warc</target_result_ext>
    <host_name>https://www.kasa.go.kr</host_name>
    <active>1</active>
    <publish>1</publish>
  </DATA_RECORD>
  <DATA_RECORD>
    <all_target_cnt>0</all_target_cnt>
    <crawl_status>30</crawl_status>
    <crawl_status_detail>33</crawl_status_detail>
    <exclude_target_cnt>0</exclude_target_cnt>
    <fail_target_cnt>0</fail_target_cnt>
    <success_target_cnt>0</success_target_cnt>
    <target_size>213</target_size>
    <target_work_cnt>0</target_work_cnt>
    <crawl_end_at>2025-07-31 08:10:32.105 +0900</crawl_end_at>
    <crawl_start_at>2025-07-31 08:08:32.105 +0900</crawl_start_at>
```

<그림 105> 화면에 출력되는 데이터를 XML로 저장(쿼리 수행)

· XML 파일 제일 상단에 화면 재현 정보가 담긴 XSLT파일 링크
(XSLT파일 생성은 아래의 기술)

```
<?xml version="1.0" encoding="UTF-8"?>
<?xml-stylesheet type="text/xsl" href="./오아시스-기본인적.xsl"?>

<개인인사기본>
  <기본인적>
    <개인기본>
      <대표개인번호>[redacted]</대표개인번호>
      <성명>[redacted]</성명>
      <주민등록번호1>[redacted]</주민등록번호1>
      <주민등록번호2>[redacted]</주민등록번호2>
      <한문명>[redacted]</한문명>
      <영문명>[redacted]</영문명>
      <성별>남자</성별>
      <국적>대한민국</국적>
      <생년월일>[redacted]</생년월일>
      <음력구분>양력</음력구분>
      <이메일>dmyang@jbnu.ac.kr</이메일>
      <건물명></건물명>
      <호실명>310호</호실명>
      <사무실전화번호>0632703249</사무실전화번호>
      <대표보직사무실전화번호></대표보직사무실전화번호>
      <자택전화번호></자택전화번호>
      <휴대폰>[redacted]</휴대폰>
      <FAX>0632703260</FAX>
      <주소1>54896</주소1>
      <주소2>전라북도 전주시 덕진구 백제대로 567 (금암동)</주소2>
      <상세주소>[redacted]</상세주소>
      <영문주소></영문주소>
      <영문상세주소></영문상세주소>
    </개인기본>
```

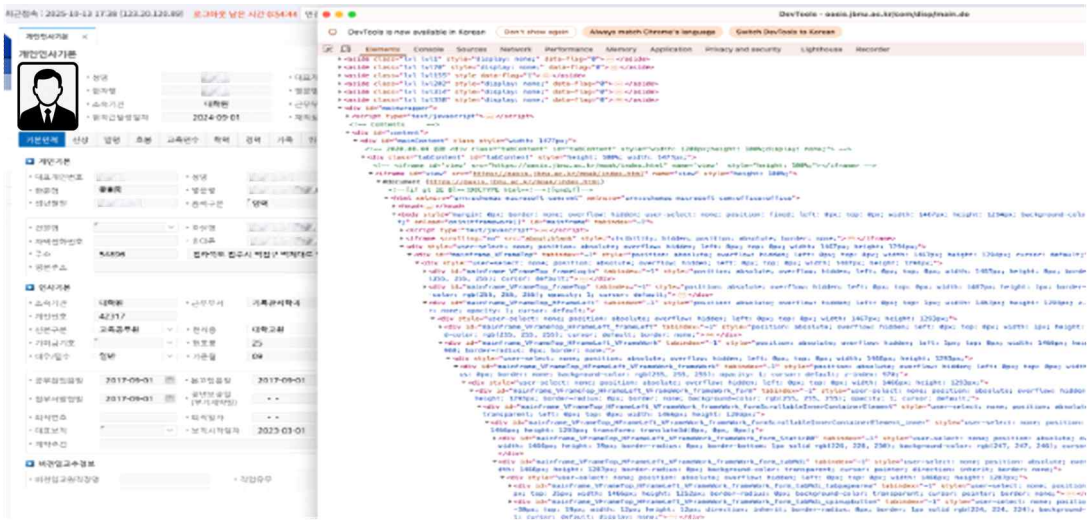
<그림 106> XML내에 XSLT 파일 선언 예시

- HTML/CSS 추출

· 원본 화면의 화면 소스 코드(HTML/CSS)에 접근이 가능한 경우 직접 추출

· 브라우저의 개발자도구(예시 : 크롬 브라우저의 개발자 도구)를 활용하여 HTML과 CSS 추출 가능

<그림 107> 브라우저 상에서 원본 화면 접속



<그림 108> 브라우저의 개발자 도구를 활용하여 HTML/CSS 파일 추출

- HTML을 XSLT 포맷으로 변환하고 XML데이터를 맵핑
  - · HTML을 XML 포맷의 XSL 파일로 변환(상단에 XML 선언 및 xsl:stylesheet, xsl:template tag 삽입)
  - · 변환 후에 Well-formed XML 체크 필요

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
   <xsl:template match="/">

<html>

<head>
    <meta http-equiv="X-UA-Compatible" content="IE=Edge" />
    <meta http-equiv="content-type" content="application/xhtml+xml; charset=UTF-8" />
    <meta name="viewport" content="user-scalable=0,target-densitydpi=device-dpi" />
    <script type="text/javascript">

    </script>
    <script type="text/javascript" src="./nexacro14lib/framework/Framework.js"></script>

    <script type="text/javascript" src="./nexacro14lib/component/EcoLibrary.js"></script>
    <script type="text/javascript" src="./nexacro14lib/component/ExtComp.js"></script>
    <script type="text/javascript" src="./nexacro14lib/component/rMateChartH5.js"></script>
    <script type="text/javascript" src="./nexacro14lib/component/JbnuNexacroLibLoad.js"></script>
    <script type="text/javascript" src="./nexacro14lib/component/UbiViewer.js"></script>

    <script type="text/javascript" src="./nexacro14lib/component/CryptoJS/jquery-1.11.0.min.js"></script>
    <script type="text/javascript" src="./nexacro14lib/component/CryptoJS/jquery-migrate-1.2.1.min.js"></scr
ipt>
    <script type="text/javascript" src="./nexacro14lib/component/CryptoJS/core-min.js"></script>
    <script type="text/javascript" src="./nexacro14lib/component/CryptoJS/aes.js"></script>
    <script type="text/javascript" src="./nexacro14lib/component/CryptoJS/pad-zeropadding.js"></script>

    <script type="text/javascript" src="./oasis.xadl.js"></script>
```

<그림 109> 추출된 HTML 파일을 XML형태의 XSLT파일로 변환

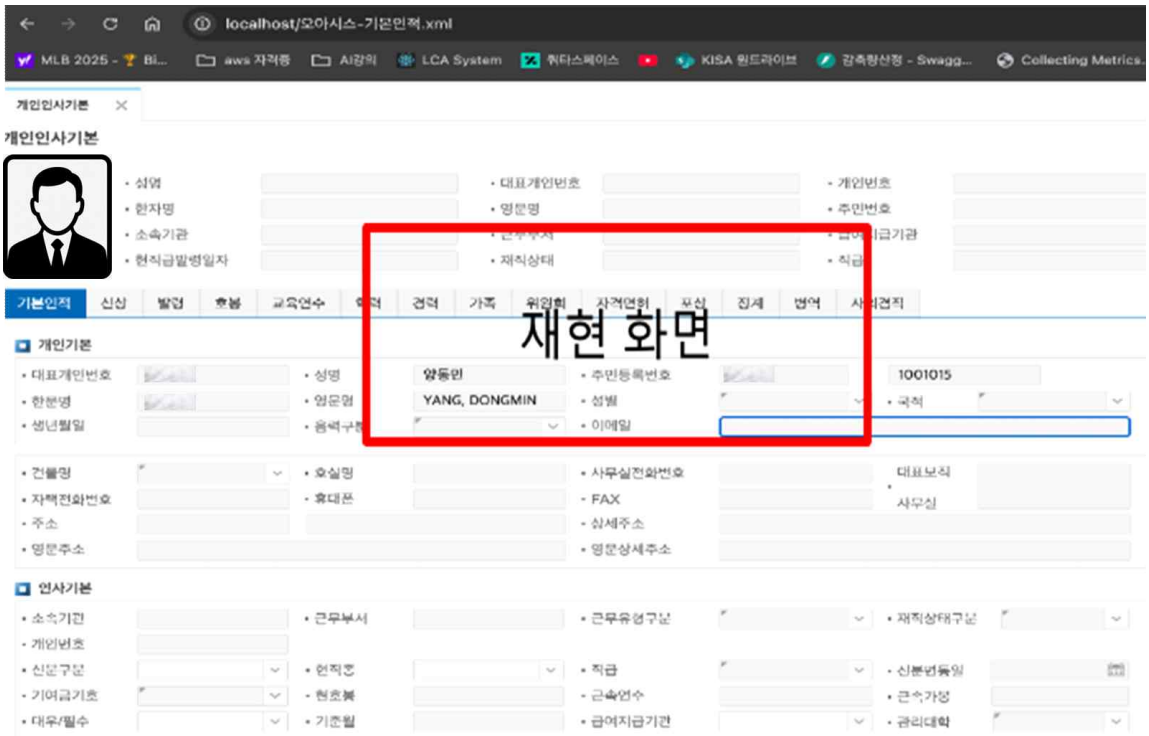
· 화면 데이터는 XML의 데이터를 참조하도록 XSLT 파일 내에 바인딩 작업 진행 (<xsl:value-of> tag 활용)

```
<tr class="temp-detail-text" >
  <td class="temp-detail-head"    >성 명 </td>
  <td    ><xsl:value-of select="개 인 기 본 /성 명 "/></td>
  <td class="temp-detail-head"    >성 별 </td>
  <td  colspan="3">
  <xsl:value-of select="개 인 기 본 /성 별 "/>
  </td>
</tr>
```

<그림 110> 브라우저의 개발자 도구를 활용하여 HTML/CSS 파일 추출

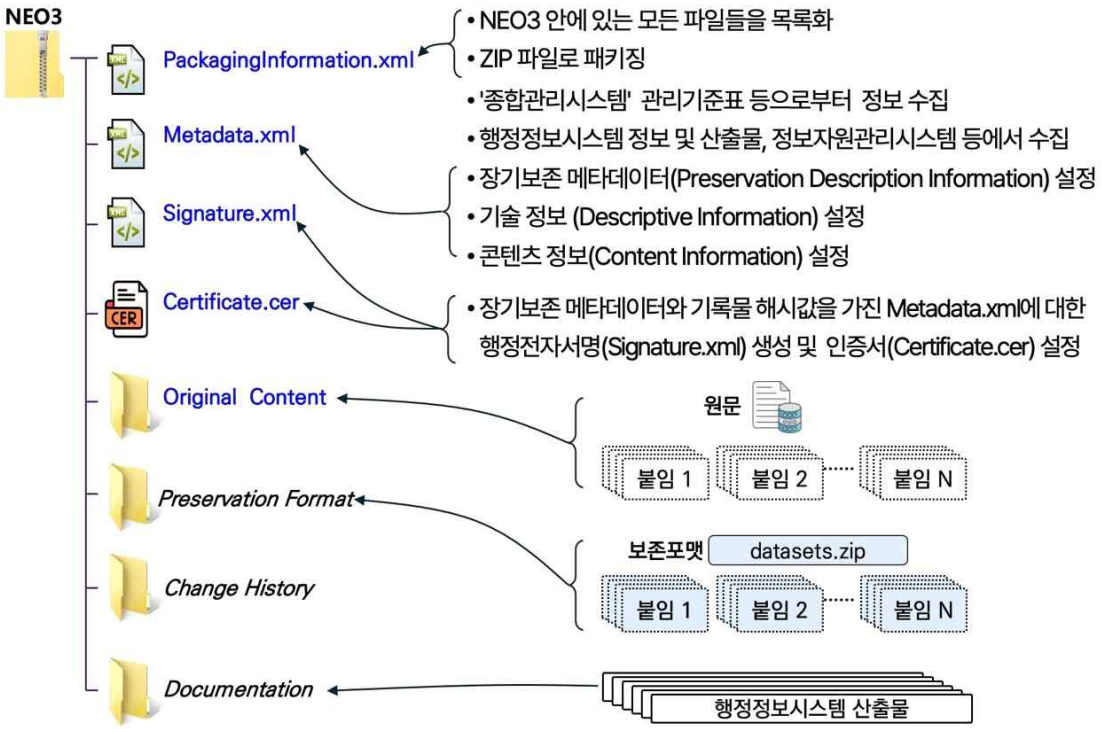
- 웹서버를 이용한 재현

· 웹서버가 설치된 환경에서 data XML을 브라우저에서 호출하면 원본 화면과 동일하게 재현됨을 확인 가능(local PC에서도 재현 파일 확인 가능)



<그림 111> 브라우저를 활용하여 local에 저장된 xml파일의 재현 예시

라. 장기보존패키지 생성 및 검증



<그림 112> 장기보존패키지 생성 및 검증 방안 개요

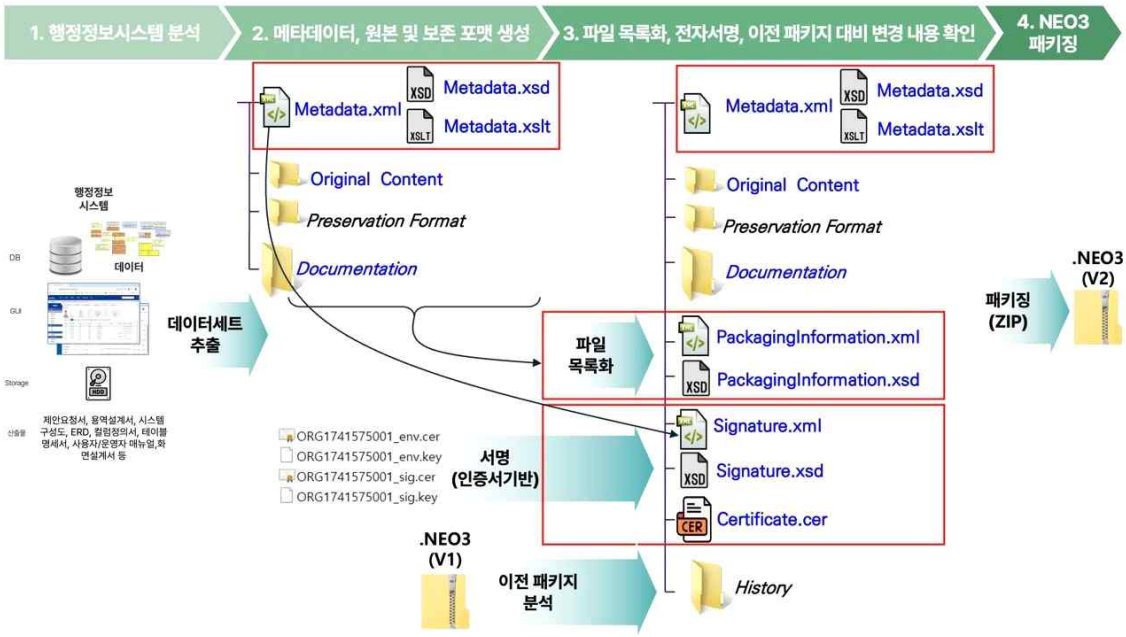
○ 장기보존패키지(NEO3) 검증을 위한 테스트베드는 설계 안에 따라 ① 행정정보시스템 분석을 통한 메타데이터와 원문·보존포맷·붙임 확보, ② 메타데이터·원문·보존포맷 등 폴더 배치, ③ 파일목록작성·전자서명 적용·히스토리 생성 ④ 패키징으로 구성함

① 행정정보시스템 분석을 통한 메타데이터와 원문·보존포맷·붙임 확보 : 대상 행정정보시스템으로부터 메타데이터, 원문, 보존포맷, 붙임파일, 재현정보, 시스템 산출물 확보

② 메타데이터·원문·보존포맷 등 폴더 배치 : 폴더 ROOT에 Metadata 파일들과 Original Format(원문 파일들) 폴더와 Preservation Format(보존포맷 파일들) 폴더를 배치

③ 파일목록 작성·전자서명 적용·히스토리 생성 : 폴더에 배치된 모든 파일에 대한 목록을 PackaginInformation.xml에 기록하고 폴더에 추가, 행정전자서명 인증관리센터에서 발급받은 기관 인증서를 이용하여 Metadata.xml에 대한 전자서명값 생성하여 Signature.xml에 서명값과 인증서 정보를 기록한 후에 폴더에 추가, 이전 버전의 NEO3 패키지가 있다면 현재 버전에서 바뀌는 파일에 대해서만 History 폴더에 추가하여 배치

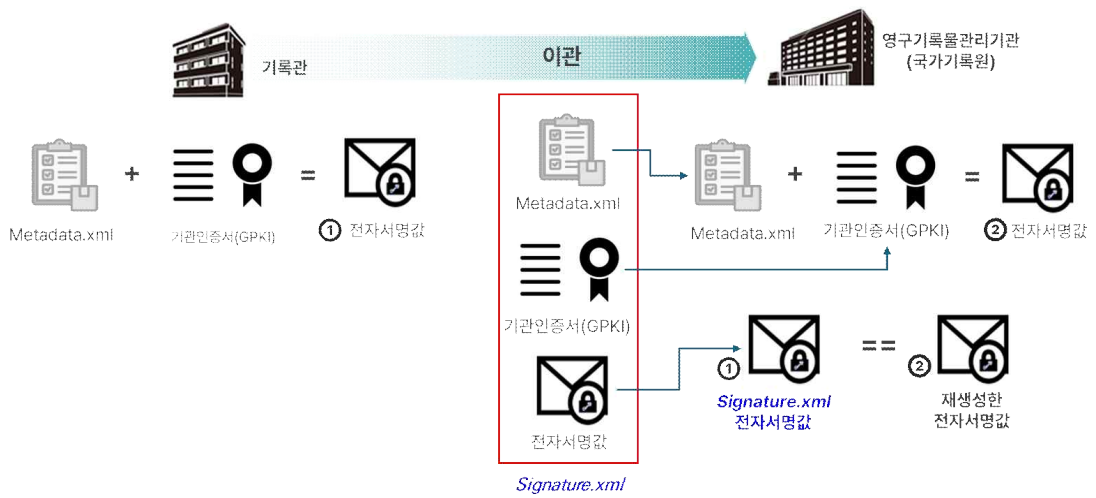
④ 패키징 : ZIP 규약을 이용하여 폴더 상의 모든 파일을 압축, 패키징하여 확장자를 .NEO3로 저장



<그림 113> 장기보존파일 NEO3 패키징 과정

○ 실데이터 기반 장기보존패키지(NEO3) 검증

- 기록관에서 발행받은 인증서를 통해 Metadata.xml의 전자서명값(①)을 생성하여 Signature.xml에 기록
- 영구기록물관리기관으로 이관된 후에 Metadata.xml 의 전자서명값(②) 을 다시 생성한 후에 ①과 ②의 값이 동일한지 비교



<그림 114> 장기보존패키지 서명 생성 및 진본성 검증 프로세스

마. 현재 운영중인 행정정보시스템에 대한 보존포맷 생성 및 검증 작업 수행

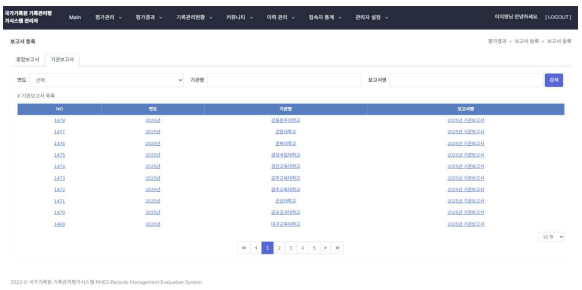
○ 현장 검증 대상 시스템

- 개요

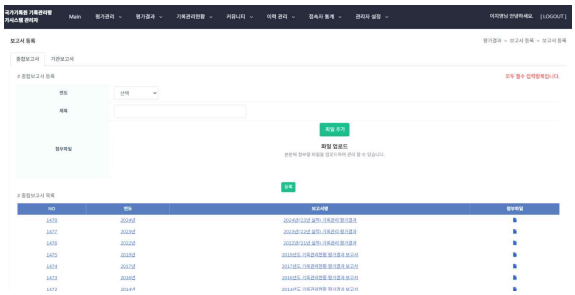
· 공공기관의 기록물 관리 수준을 체계적으로 평가하고 개선하기 위해 국가기록원에서 운영중인 **기록관리평가시스템에 대해서 보존포맷 생성 및 검증 작업 직접 수행**

- 작업 내역

· 해당 시스템에서 **기관보고서와 종합보고서 화면에서 사용중인 모든 테이블에 대해서 original type의 보존포맷 생성**



<그림 115> 기관보고서 화면



<그림 116> 종합보고서 화면

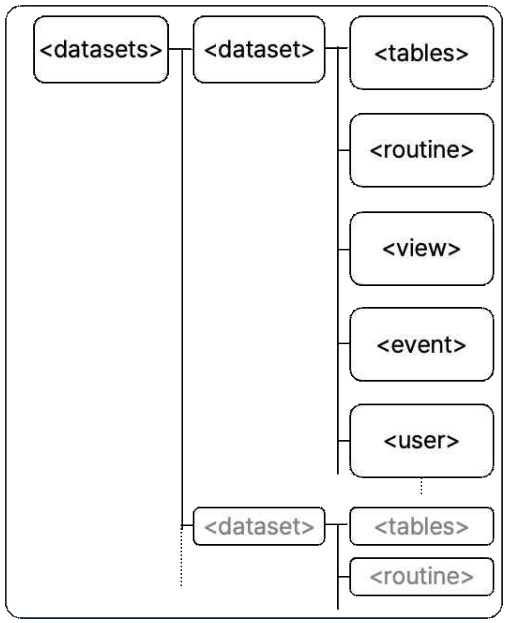
- **기관평가 결과 상세화면에 대한 derived type의 보존포맷 생성 및 재현 파일 생성**

<그림 117> 기관평가 상세화면

○ 보존포맷 생성 과정

- 개요

· 기 정의된 metadata.xml 규격에 따라서 기록관리평가시스템의 DBMS의 table 정보, routine 정보, view 정보, event 정보, user 정보를 매칭되는 tag에 기술하는 작업 현장 수행



<그림 118> metadata.xml의 구조

- **테이블별 data xml 파일 이름 수정 및 metadata.xml 맵핑 작업 수행**

① 기관보고서와 관련된 총 10개의 테이블에 대한 DDL문과 각 테이블의 XML 데이터를 metadata.xml 파일에 맵핑하는 작업 수행

· 기관보고서와 관련된 테이블의 xml 데이터를 table1.xml ~ table10.xml로 renaming후에 content/table1~table10 폴더 하위에 위치시키고 metadata.xml내에 table element의 datapath attribute에 맵핑하는 작업 수행(**table element의 soruceType attribute는 "origianl"로 설정**)

· 각 테이블의 DDL문을 table/sources tag 하위에 기록(CDATA활용)

```
▼<dataset unitFunctionName="기관보고서정보" unitFunctionID="REPORT_01">
  ▼<tables>
    ▼<table datapath="./content/table1/table1.xml" name="EVL_IDX" sourceType="original">
      ▼<sources>
          <![CDATA[ -- TIBERO.EVL_IDX definition -- Drop table -- DROP TABLE TIBERO.EVL_IDX; CREATE TABLE TIBERO.EVL_IDX ( EVL_I
          OTHBC_AT VARCHAR(1) NOT NULL, MESURE_MTH VARCHAR(4000), MESURE_STDR VARCHAR(4000), QESITM_CO NUMBER(3,0), REGISTER_ID
          DEFAULT 'N', ADDPNT NUMBER(5,2) DEFAULT 0, IMPRVM_REQUST_MATTER VARCHAR(4000), "YEAR" VARCHAR(4), CONSTRAINT EVL_IDX_
        </sources>
      </table>
    ▼<table datapath="./content/table2/table2.xml" name="EVL_IDX_GROUP_QESITM" sourceType="original">
      ▼<sources>
          <![CDATA[ -- TIBERO.EVL_IDX_GROUP_QESITM definition -- Drop table -- DROP TABLE TIBERO.EVL_IDX_GROUP_QESITM; CREATE TA
          GROUP_QESITM_NM VARCHAR(150), QESITM_NM VARCHAR(150), QESITM_SE VARCHAR(10) DEFAULT '선택형', QESITM_CO NUMBER(3,0), R
          EVL_IDX_GROUP_QESITM_PK PRIMARY KEY (EVL_IDX_ID,GROUP_QESITM_ID,QESITM_ID) ); ]]>
        </sources>
      </table>
    ▼<table datapath="./content/table3/table3.xml" name="EVL_LAST_RSLT" sourceType="original">
      ▼<sources>
          <![CDATA[ -- TIBERO.EVL_LAST_RSLT definition -- Drop table -- DROP TABLE TIBERO.EVL_LAST_RSLT; CREATE TABLE TIBERO.EVL
          NULL, EVL_KND_CD VARCHAR(8) NOT NULL, QESITM_ID VARCHAR(10), EVL_STEP VARCHAR(1) NOT NULL, EVL_SCORE NUMBER(5,2), EVL_
          DEFAULT SYSDATE, REGIST_TIME TIME(6), UPDUSR_ID VARCHAR(8), UPDT_DT DATE DEFAULT SYSDATE, EVL_STTUS_CD VARCHAR(8), EVL
          (EVL_IDX_ID,EVL_KND_CD,EVL_PLAN_ID,EVL_STEP,EVL_TRGT_GROUP_CD,GROUP_QESITM_ID,INSTT_ID) ); ]]>
        </sources>
      </table>
```

<그림 119> 기관보고서에 대한 metadata.xml 작성

② 종합보고서와 관련된 총 2개의 테이블에 대한 DDL문과 각 테이블의 XML 데이터를 metadata.xml 파일에 맵핑하는 작업 수행

· 기관보고서와 관련된 테이블의 xml 데이터를 table11.xml ~ table12.xml로 renaming후에 content/table11~table12 폴더 하위에 위치시키고 metadata.xml내에 table element의 "datapath" attribute에 맵핑하는 작업 수행(table element의 soruceType attribute는 "origianl"로 설정)

· 각 테이블의 DDL문을 table/sources tag 하위에 기록(CDATA활용)

```
<dataset unitFunctionName="종합보고서정보" unitFunctionID="REPORT_02">
  <tables>
    <table datapath="./content/table11/table11.xml" name="EVL_REPORT" sourceType="original">
      <sources>
        <![CDATA[ -- TIBERO.EVL_REPORT definition -- Drop table -- DROP TABLE TIBERO.EVL_REPORT; CREATE TABLE TIBERO.EVL_REPORT ( REPORT_I
        FILE_ID NUMBER, DELETE_AT VARCHAR(1) DEFAULT 'N', REGISTER_ID VARCHAR(10), REGIST_DT DATE DEFAULT SYSDATE, UPDUSR_ID VARCHAR(8), L
      </sources>
    </table>
    <table datapath="./content/table12/table12.xml" name="EXIT_ATCHMNFL" sourceType="original">
      <sources>
        <![CDATA[ -- TIBERO.EXIT_ATCHMNFL definition -- Drop table -- DROP TABLE TIBERO.EXIT_ATCHMNFL; CREATE TABLE TIBERO.EXIT_ATCHMNFL (
        ATCHMNFL_REAL_NM VARCHAR(255) NOT NULL, ATCHMNFL_SIZE NUMBER(11,0), ATCHMNFL_TY VARCHAR(255), ATCHMNFL_EXTSN VARCHAR(10), ATCHMNFL
        (ACC_SN,ACC_TY,ATCHMNFL_ORDR) ); CREATE INDEX EXIT_ATCHMNFL_IDX1 ON TIBERO.EXIT_ATCHMNFL (REGIST_DE); ]]>
      </sources>
    </table>
  </tables>
</dataset>
```

<그림 120> 종합보고서에 대한 metadata.xml 작성

③ 기관평가 결과 상세화면에 대한 XML 데이터와 관련 테이블의 DDL문을 metadata.xml 파일에 맵핑하는 작업 수행

· 기관평가 결과 상세화면의 재현에 필요한 데이터인 항목별 점수, 점수 합계, 각 기관의 최종 평가 등급에 대한 xml 데이터를 table13.xml ~ table15.xml로 renaming후에 content/table13~table15 폴더 하위에 위치시키고 metadata.xml내에 맵핑하는 작업 수행 (table element의 datapath attribute에 xml파일 위치 표시, table element의 soruceType attribute는 “derived”로 설정)

· 각각의 xml 데이터를 추출하는데 사용했던 쿼리 상에서 사용하는 모든 테이블에 대한 DDL문을 table/sources tag 하위에 모두 기록(CDATA 활용)

· table>representation>location tag에 재현 파일들의 경로를 표시(xslt 파일을 representation/table13_style.xsl로 명칭)

```
<dataset unitFunctionName="기록관리 기관평가결과" unitFunctionID="EVAL_01">
  <tables>
    <table datapath="./content/table13/table13.xml" name="기록관리 기관평가결과_화면_강릉원주대학교_항목별점수" sourceType="derived">
      <derivation method="SQL">
        <![CDATA[ SELECT EVL_IDX_SE_NM , EVL_IDX_IEM_NM , EVL_IDX_NM , ALLOT , EVL_SCORE , ALL_AVG , EVL_SCORE - ALL_AVG AS AM , EVL_OPI
        Z WHERE CODE_ESTBS_ID = 'INDX_TC' AND EVL_IDX_SE_CODE = Z.CODE_ID) AS EVL_IDX_SE_NM , (SELECT CODE_NM FROM EXIT_CODE Z WHERE COD
        E.EVL_OPINION , D.EVL_IDX_IEM_CODE , D.EVL_IDX_SE_CODE , D.EVL_IDX_ID , D.EVL_IDX_NM , (SELECT GROUP_QESITM_NM FROM EVL_IDX_GROU
        ROWNUM=1 ) AS GROUP_QESITM_NM , (SELECT ROUND(AVG(DECODE(EVL_LAST_RSLT.EVL_SCORE,'-99',NULL,EVL_LAST_RSLT.EVL_SCORE)),'1') FROM
        EVL_LAST_RSLT.EVL_IDX_ID =A.EVL_IDX_ID AND EVL_LAST_RSLT.GROUP_QESITM_ID =A.GROUP_QESITM_ID AND EVL_KND_CD='ESTM0003' AND EVL_ST
        QESITM_SE = '누적형' THEN SUM(ALLOT) ELSE MAX(ALLOT) END AS ALLOT FROM EVL_PLAN_IDX_QESITM_ALLOT SUA ,(SELECT EVL_IDX_ID,GROUP_Q
        SUB.EVL_IDX_ID AND SUA.GROUP_QESITM_ID = SUB.GROUP_QESITM_ID GROUP BY EVL_PLAN_ID, EVL_TRGT_GROUP_CD, QESITM_SE, SUA.EVL_IDX_ID
        EVL_PLAN_ID,EVL_TRGT_GROUP_CD,INSTT_ID,EVL_KND_CD,EVL_IDX_ID,GROUP_QESITM_ID , SUM(EVL_SCORE) AS EVL_SCORE , LISTAGG(EVL_OPINION
        EVL_KND_CD ='ESTM0003' AND EVL_STEP ='2' AND EVL_SCORE != -99 GROUP BY EVL_PLAN_ID,EVL_TRGT_GROUP_CD,INSTT_ID,EVL_KND_CD,EVL_IDX
        B.EVL_TRGT_GROUP_CD AND A.EVL_IDX_ID = D.EVL_IDX_ID AND A.EVL_PLAN_ID = E.EVL_PLAN_ID AND A.EVL_TRGT_GROUP_CD = E.EVL_TRGT_GROUP
        B.INSTT_ID = #{insttId} AND C.DELETE_AT = 'N' ) ORDER BY TO_NUMBER(EVL_IDX_ORDR) ]]>
      </derivation>
      <sources>
        <![CDATA[ -- TIBERO.EXIT_CODE definition -- Drop table -- DROP TABLE TIBERO.EXIT_CODE; CREATE TABLE TIBERO.EXIT_CODE ( CODE_ESTB
        NUMBER(3,0) DEFAULT 1 NOT NULL, USE_AT VARCHAR(1) NOT NULL, ASSTN_CODE_ID VARCHAR(8), CONSTRAINT EXIT_CODE_PK PRIMARY KEY (CODE_
        TABLE TIBERO.EVL_IDX_GROUP_QESITM ( EVL_IDX_ID VARCHAR(10) NOT NULL, GROUP_QESITM_ID VARCHAR(8) NOT NULL, QESITM_ID VARCHAR(8) N
        REGISTER_ID VARCHAR(8), REGIST_DE DATE DEFAULT SYSDATE, REGIST_TIME TIME(6), UPDUSR_ID VARCHAR(8), UPDT_DT DATE DEFAULT SYSDATE,
        Drop table -- DROP TABLE TIBERO.EVL_LAST_RSLT; CREATE TABLE TIBERO.EVL_LAST_RSLT ( EVL_PLAN_ID VARCHAR(10) NOT NULL, EVL_IDX_ID
        VARCHAR(10), EVL_STEP VARCHAR(1) NOT NULL, EVL_SCORE NUMBER(5,2), EVL_OPINION VARCHAR(4000), ADDPNT NUMBER(5,2) DEFAULT 0, ATCHM
        VARCHAR(8), UPDT_DT DATE DEFAULT SYSDATE, EVL_STTUS_CD VARCHAR(8), EVL_TRGT_GROUP_CD VARCHAR(8) NOT NULL, QESITM_SELECT_AT VARCH
        (EVL_IDX_ID,EVL_KND_CD,EVL_PLAN_ID,EVL_STEP,EVL_TRGT_GROUP_CD,GROUP_QESITM_ID,INSTT_ID) ); -- TIBERO.EVL_PLAN_IDX_QESITM_ALLOT d
        VARCHAR(10) NOT NULL, EVL_TRGT_GROUP_CD VARCHAR(8) NOT NULL, EVL_IDX_ID VARCHAR(10) NOT NULL, GROUP_QESITM_ID VARCHAR(8) NOT NUL
        SYSDATE, REGIST_TIME TIME(6), UPDUSR_ID VARCHAR(8), UPDT_DT DATE DEFAULT SYSDATE, EVL_IDX_ORDR VARCHAR(10), CONSTRAINT EVL_PLAN_
        TIBERO.EVL_IDX_GROUP_QESITM definition -- Drop table -- DROP TABLE TIBERO.EVL_IDX_GROUP_QESITM; CREATE TABLE TIBERO.EVL_IDX_GROU
        VARCHAR(150), QESITM_NM VARCHAR(150), QESITM_SE VARCHAR(10) DEFAULT '선택형', QESITM_CO NUMBER(3,0), REGISTER_ID VARCHAR(8), REG
        EVL_IDX_GROUP_QESITM_PK PRIMARY KEY (EVL_IDX_ID,GROUP_QESITM_ID,QESITM_ID) ); ]]>
      </sources>
      <representationInfo method="xml_xslt">
        <description> XML과 XSLT를 이용해 table13 데이터를 HTML로 재현하는 방식 </description>
        <location>./representation/table13/table13_style.xsl</location>
        <location>./representation/table13/css</location>
      </representationInfo>
    </table>
    <table datapath="./content/table14/table14.xml" name="기록관리 기관평가결과_화면_강릉원주대학교_합계" sourceType="derived">
      <derivation method="SQL">
        <![CDATA[ SELECT sum(allot) AS SUM_ALLOT ,sum(EVL_SCORE) AS SUM_EVL_SCORE ,sum(ALL_AVG) AS SUM_ALL_AVG ,sum(rm) AS SUM_RM FROM(
        EVL_OPINION , EVL_IDX_IEM_CODE , EVL_IDX_SE_CODE , EVL_IDX_ID FROM( SELECT (SELECT CODE_NM FROM EXIT_CODE Z WHERE CODE_ESTBS_ID
```

④ 기관평가 결과 상세화면에 대한 xslt 재현 파일을 생성(table13_style.xsl)하고 xslt 재현파일과 data xml 파일을 맵핑하는 작업 수행

· 기관평가 결과 상세화면의 소스 파일인 html 파일을 xml 형태의 xslt파일로 변환(xml tag 삽입 및 well-formed xml 체크 필요)

```
<?xml version="1.0" encoding="UTF-8"?>
<xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
    <xsl:template match="/">

<xsl:variable name="sub1" select="document('./table31.xml')"/>
<xsl:variable name="sub2" select="document('./table32.xml')"/>
<xsl:variable name="sub3" select="document('./table33.xml')"/>

<html>
<head>
<meta charset="UTF-8"/>
<title>보고서 등록 | 국가기록원 기록관리 시스템 관리자</title>
<meta name="viewport" content="width=device-width, initial-scale=1.0"/>
<meta content="Premium Multipurpose Admin &amp; Dashboard Template" name="description"/>
<meta content="Themesdesign" name="author"/>
<link href="./css/bootstrap.css" id="bootstrap-style" rel="stylesheet"/>
<link href="./css/icons.min.css" rel="stylesheet"/>
<link href="./css/icons.css" rel="stylesheet"/>
<link href="./css/app.css" rel="stylesheet"/>
<link href="./css/app.min.css" id="app-style" rel="stylesheet"/>
<link href="/resource/cmm/css/loding.css" rel="stylesheet"/>
<link href="./css/exit_content.css" rel="stylesheet"/>

<script src="/resource/adm/libs/jquery/jquery.min.js"></script>
<script src="/resource/cmm/js/jquery_utils.js"></script>

</head>
<body data-topbar="dark" data-layout="horizontal" style="overflow-y: auto;">
        <div id="layout-wrapper">

        <header id="page-topbar">
            <div class="navbar-header">
                <div class="d-flex">

                    <!-- 모바일 메뉴 활성 창 -->
                    <button type="button" class="btn btn-sm px-3 font-size-24 d-lg-none header-item"
                        <i class="ri-menu-2-line align-middle"></i>
```

<그림 122> 기관평가 결과 상세화면 xslt 파일

· xslt파일 내부에 관련된 data xml 파일 맵핑(table13.xml ~table15.xml에 대해서 변수 선언)

- <xsl:variable name="sub1" select="document('../content/table13/table13.xml')">
- <xsl:variable name="sub2" select="document('../content/table14/table14.xml')">
- <xsl:variable name="sub3" select="document('../content/table15/table15.xml')">

· xslt파일에서 화면에 display 되어야 할 xml 데이터들 맵핑 작업 수행

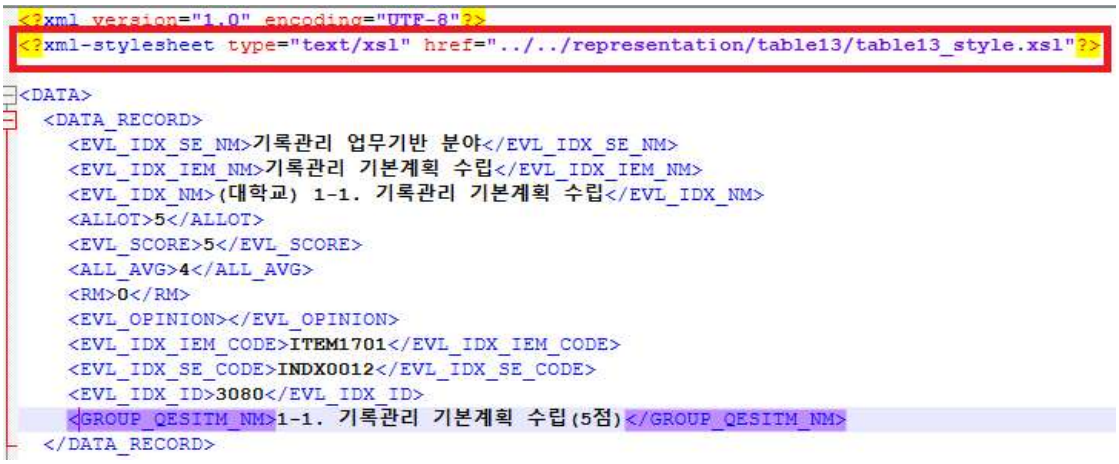
```
<tr>
    <td style="display: none;">기록관리 업무기반 분야</td>
    <td style="text-align: left;">(공공)전담조직·인력 배치 및 업무분장의 적절성</td>
    <td style="text-align: left;">(대학교) 1-3. 전담조직 인력 배치 및 업무분장의 적절성</td>
    <td style="text-align: left;">1-3-나.  기록물관리 전문요원 업무분장의 적절성(5점)</td>
    <td style="background-color: #DCE6F1;">5</td>
    <td style="background-color: #EBF1DE;color:#537296;"><xsl:value-of select="$sub1/DATA/DATA_RECORD[GROUP_QESITM_NM = '1-3-나.  기
    <td><xsl:value-of select="$sub1/DATA/DATA_RECORD[GROUP_QESITM_NM = '1-3-나.  기록물관리 전문요원 업무분장의 적절성(5점)']/ALL_AVG"/></
    <td><xsl:value-of select="$sub1/DATA/DATA_RECORD[GROUP_QESITM_NM = '1-3-나.  기록물관리 전문요원 업무분장의 적절성(5점)']/RM"/></td>
</tr>

<tr>
    <td style="display: none;">기록관리 업무기반 분야</td>
    <td style="text-align: left;">(공공)기록관리 교육 실시 및 이수</td>
    <td style="text-align: left;">(대학교) 1-4. 기록관리 교육 실시 및 이수</td>
    <td style="text-align: left;">1-4-다. 기록관리 담당자 교육 이수 여부(3점)</td>
    <td style="background-color: #DCE6F1;">3</td>
    <td style="background-color: #EBF1DE;color:#537296;"><xsl:value-of select="$sub1/DATA/DATA_RECORD[GROUP_QESITM_NM = '1-4-다. 기록
    <td><xsl:value-of select="$sub1/DATA/DATA_RECORD[GROUP_QESITM_NM = '1-4-다. 기록관리 담당자 교육 이수 여부(3점)']/ALL_AVG"/></td>
    <td><xsl:value-of select="$sub1/DATA/DATA_RECORD[GROUP_QESITM_NM = '1-4-다. 기록관리 담당자 교육 이수 여부(3점)']/RM"/></td>
</tr>
```

<그림 123> 기관평가 결과 상세화면에서 항목별 점수에 대한 xml 데이터 맵핑

```
<tbody style="font-size:13px;">
    <tr style="text-align: center; font-size: 2em;font-weight: bold;">
        <td colspan="8" style="background-color: #4F81BD;color:black;">
            2025년 기록관리 기관평가 결과
        </td>
    </tr>
    <tr>
        <td rowspan="2" colspan="4" style="font-weight: bold;text-align: center; font-size: 1.8em;vertical-align: middle;">
            강릉원주대학교
        </td>
        <td colspan="2" style="text-align: center;font-size:1.2em;background-color: #B8CCE4;font-weight: bold;">등급</td>
        <td colspan="2" style="text-align: center;font-size:1.2em;background-color: #B8CCE4;font-weight: bold;">총점</td>
    </tr>
    <tr>
        <td colspan="2" style="text-align: center;font-size:1.2em;font-weight: bold;"><xsl:value-of select="$sub3/DATA/DATA_RECORD[1]/LAST_EVL_UPDT_GRAD"/></td>
        <td colspan="2" style="text-align: center;font-size:1.2em;font-weight: bold;"><xsl:value-of select="$sub3/DATA/DATA_RECORD[1]/LAST_EVL_UPDT_SCORE"/></td>
    </tr>
    <tr>
        <td colspan="8" style="text-align: center;font-size:1.4em;font-weight: bold;color: white;background-color: #4F81BD;">2025년 기록관리 기관평가 지표별 세부 점수</td>
    </tr>
```

<그림 124> 기관평가 결과 상세화면에서 등급에 대한 xml 데이터 맵핑

· 데이터가 포함된 xml(table13.xml)파일 내부에 위에서 생성한 xslt 링크 표시



<그림 125> 데이터가 포함된 xml 파일에 생성한 xslt화면 선언

· 웹서버를 설치하고 웹서버 상에서 main이 되는 data.xml(content/data13.xml)을 호출하여 정상 재현 확인

· 화면에서 사용하는 첨부파일이 존재할 경우 attachment/ 폴더에 첨부파일

<그림 126> 웹서버를 활용한 정상 재현 확인

· 작업이 모두 완료된 후에 root 폴더인 DatasetPF 하위의 모든 파일 목록을 manifest.xml에 기록하고 전체 폴더를 zip으로 압축하여 단일 파일 형태의 보존포맷으로 저장

<그림 127> manifest.xml에 모든 파일 기록 <그림 128> 최종적으로 생성된 보존포맷 폴더 구조

제4장 총괄연구개발과제의 연구결과 고찰 및 결론

4.1 데이터세트 보존포맷 및 장기보존패키지의 현황 및 적합성 조사 및 분석

가. 국내외 아카이브 기관 데이터세트 보존포맷 및 장기보존패키지의 현황 조사 및 분석

○ (주요연구결과) 국내외 아카이브 기관 조사 결과, 오픈 포맷(XML, RDF, URI)과 국제표준(Unicode, SQL) 기반 설계가 상호운용성 확보에 효과적임. 미 의회도서관·유로피아나·민호 대학 사례처럼 URI로 외부 저장소 대용량 파일을 메타데이터에 연결하는 방식이 보존·접근 유연성 높이는 핵심 전략으로 나타남. 민호 대학은 SIARD2 선별적 보존 기능을 DBPTK에 구현해 실무 활용 가능성 입증함

○ (향후 계획 제안) 향후 연구는 오픈 포맷·국제표준 기반 보존체계 강화 필요함. 보존포맷 설계 시 URI 활용해 외부 저장소 대용량 파일을 메타데이터와 연결하는 방안 도입 필요함. 이를 통해 통합관리와 시스템 확장성, 장기보존 인프라 강화 가능함

나. 기록분야 이외 데이터세트 보존포맷 및 장기보존패키지의 현황 조사 및 분석

○ (주요연구결과) 기업 분야는 규제 준수 중심으로 SAP 등 솔루션이 DB·파일 패키징 보존 지원, OAIS 기반 대용량 저장 가능함. 과학기술 분야는 FAIR 원칙 하에 오픈 포맷과 전용 표준을 활용해 데이터·분석 환경 통합 패키징함

○ (향후 계획 제안) 오픈 포맷과 국제표준 기반 설계로 상호운용성 확보 예정임. URI를 활용해 외부 대용량 파일 연결 방안 도입하고, PB급 파일과 DB를 논리·물리적으로 패키징하는 구조 반영함. 재현 가능한 보존체계 기반 설계 강화 목표임

4.2 데이터세트(DB형) 유형의 기록물에서의 원문에 대한 이해

가. 디지털화(Digitization) 또는 변환(Migration)에 따른 원문 폐기 근거 해외 법령 및 표준

○ (주요연구결과) 디지털화의 경우, 미국 NARA와 호주 PROV 모두 일정한 기준을 충족할 때 디지털화 완료 이후 원본기록물의 폐기를 허용하는 경향을 보임. 호주 PROV와의 서면 질의 결과, 실제로 디지털화 이후 원본기록물 폐기 허가 요청이 접수되어 승인된 사례가 존재함. 반면 전자기록물 간 매체 변환(Migration)의 경우에는 원본기록물의 폐기가 명시적으로 규정되어 있지 않고 간접적으로 언급되어 있어 전자기록물 관리·보존·이관과 관련된 항목을 종합적으로 분석하여 폐기 관련 맥락을 파악함

○ (향후 계획 제안) 디지털화의 경우 원문 폐기에 대한 명시적인 규정과 사례를 확인할 수 있었지만, 매체 변환(Migration)의 경우 명시적인 폐기 근거를 확인하기 어려움. 향후 전자기록물의 매체 변환(Migration) 운영 사례를 확인하고 직·간접적으로 폐기를 언급한 규정의 분석이 필요함. 이를 통해 매체 변환(Migration) 이후 원문 폐기 가능성을 검토하고, 폐기와 관련된 규정의 해석 범위와 적용 가능성을 명확히 할 필요가 있음

나. 호주 빅토리아주 PROV(Public Record Office Victoria)의 VEO3에서 제시하는 원문 포함 방식과 서면으로 알아본 데이터세트형 기록물의 관리 사례

○ (주요연구결과) 호주 빅토리아주 PROV의 「(PROV) PROS 19/05 S3: LONG TERM SUSTAINABLE FORMATS」 표준을 통해 VEO에 원문 미포함에 대한 요건을 확인함

○ (향후 계획 제안) 국내 장기보존패키지에서도 기술적·행정적·관리적 요건을 충족하는 경우에는, 대량·대용량의 원문 파일을 패키지에 포함하지 않을 수 있는 기준을 마련하고 이를 제도적으로 뒷받침할 필요가 있음

다. 디지털화 기록물 및 데이터세트형 기록물의 보존 방법에 관한 호주 빅토리아주 PROV 서면 질의 사항

○ (주요 연구결과) 호주 빅토리아주 PROV의 장기보존패키지 운영 방식과 디지털화 절차, 데이터세트(DB형) 기록관리 사례를 확인할 수 있었음

○ (향후 계획 제안) NARA와 PROV 중 PROV로부터만 답변을 받았음. NARA의 디지털화 기록물 및 데이터세트 보존 방법에 대해서도 추후 문헌조사, 서면질의 등을 통해서 진행할 필요가 있음

라. 데이터세트(DB형) 유형의 기록물의 원문에 대한 이해와 제안

○ (주요 연구결과) 미국 NARA와 호주 PROV 사례는 원문 대체를 위해 기술적 기준·검증·승인 절차가 필수이며, 데이터세트(DB형) 유형 전자기록물에 대해서는 기록물이 추구하는 본래 목적을 중심으로 원문 재정의를 고려할 필요가 있음

○ (향후 계획 제안) 국내 데이터세트(DB형) 기록관리에서도 검증된 도구 기반 추출본을 '대체 원문'으로 인정할 수 있는 기준과 절차를 마련할 필요가 있음

4.3 데이터세트(DB형) 유형의 기록물에서의 보존포맷에 대한 이해

가. 보존포맷 개요

○ (주요연구결과) 보존포맷은 전자기록물을 장기간 안전하게 보존하기 위한 파일형식으로, 단순한 형식 변환이 아닌 필수보존속성을 유지하는 변환 체계의 필요성이 확인됨

○ (향후 계획 제안) 보존포맷으로 단순히 변환하는 것만으로는 마이그레이션이 완료된 것이 아니며, 필수보존속성이 온전히 유지되어야 비로소 완료된다는 점을 기록관리 분야에 알릴 필요가 있음

나. 전자기록물(원문)의 보존포맷 변환(Migration) 과정

○ (주요연구결과) 전자기록물의 보존포맷 변환은 단순한 형식 변경이 아니라, 유형별 필수보존속성을 유지하기 위한 기술적·절차적 체계로 구성되어야 함을 확인함

○ (향후 계획 제안) 기록물 유형별 필수보존속성을 체계적으로 검증할 수 있는 표준화된 기준을 마련하고, 이를 기반으로 보존포맷 변환 과정에서 속성 손실 없이 자동으로 변환·검증할 수 있는 방안을 지속적으로 강구할 필요가 있음

다. 데이터세트 유형 전자기록물(원문)의 보존포맷 변환 과정

○ (주요연구결과) DB형 데이터세트의 보존포맷 변환은 단일 포맷이나 소프트웨어로는 필수보존속성을 완전하게 유지하기 어려워, 다중 보존포맷의 조합과 구조·관계정보를 보존하는 표준화된 변환 절차가 필요함을 확인함

○ (향후 계획 제안) DB형 데이터세트뿐만 아니라 다른 유형의 기록물에도 표준화된 변환 방식이 없다면, 필수보존속성을 온전히 유지하기 위해 여러 보존포맷을 조합한 마이그레이션 방안을 검토할 필요가 있음

4.4 데이터세트의 이관·보존을 위한 새로운 보존포맷 형태 도출 및 규격 작성

가. 데이터세트(DB형) 유형 보존포맷의 설계 원칙

○ (주요연구결과) 기존 SIARD의 한계를 보완한 오픈포맷 기반 보존포맷 설계를 통해, DB형 데이터세트의 구조·내용·무결성을 모두 보장하면서 다양한 DBMS 환경에서 호환 가능한 체계를 제시함

○ (향후 계획 제안) DB형 데이터세트뿐만 아니라 다른 유형의 기록물에도 변환을 위한 설계 원칙을 수립하는 것을 제안함

나. 데이터세트(DB형) 유형의 오픈포맷 기반 보존포맷 구조 설계

○ (주요연구결과) DB형 데이터세트 보존포맷은 메타데이터, 데이터, 붙임 파일, 재현 파일, 구성목록으로 구성되어 있으며, 각 요소가 구조적 일관성과 무결성을 유지하도록 체계적으로 설계됨

○ (향후 계획 제안) 다양한 DBMS에 대해서 DB형 데이터세트가 보유한 어려 요소들이 온전히 수용되는 구조인지 추가적인 검토가 필요함

다. 데이터세트(DB형) 유형의 보존포맷 생성 절차

○ (주요연구결과) DB형 데이터세트의 보존포맷 생성 절차는 metadata.xml, content/, attachment/, representation/, manifest.xml의 단계별 구조를 통해 데이터의 구조·내용·무결성을 체계적으로 보존하도록 설계됨

○ (향후 계획 제안) MySQL, MS SQL Server, Oracle 이외에 다양한 DBMS에 대해서 DB형 데이터세트가 보유한 여러 요소들이 온전히 생성되는 절차인지 추가적인 검토가 필요함

4.5 데이터세트 유형에 적합한 장기보존패키지(NEO3) 규격 설계

가. 장기보존패키지(NEO3) 구조 및 구성요소 정의

○ (주요연구결과) 본 연구는 「공공기록물법 시행령」과 NAK 31-2를 근거로 장기보존패키지

(NEO3)의 구조와 구성요소를 정의함. NEO3는 메타데이터, 전자서명, 원문·보존포맷 등으로 구성되며, 이를 통해 기록의 무결성과 진본성을 보장하고 장기 재현 가능성을 확보함

○ (향후 계획 제안) 두 규정에서 사용되는 용어가 서로 다르므로, 향후 용어의 통일성을 위해 재개정할 것은 제안함

나. 장기보존패키지가 보존해야 할 데이터세트 유형 기록물의 범위 정의

○ (주요연구결과) 데이터세트 유형 기록물의 보존 범위는 DB 데이터와 붙임은 필수, 재현 정보는 조건부, 주요 산출물은 선택·맥락으로 구분됨

○ (향후 계획 제안) 주요 산출물이 보존 대상 기록물인지, 아니면 기록물 이해를 위한 보조자료인지 판단하기가 쉽지 않음. 주요 산출물은 데이터세트 유형 기록물에 직접 연결되지 않고, 행정정보시스템의 부수적인 결과물로 간접적으로만 관련되므로 보존 대상에서 제외하는 것으로 제안함

다. 데이터세트 유형에 적합한 '장기보존 메타데이터' 설계

○ (주요연구결과) 데이터세트의 특성을 반영하여 장기보존패키지(NEO3)의 메타데이터를 설계함. 국가기록원의 SIP 메타정보 구조를 기반으로 요소 계층을 재구성하고, 공통 요소는 유지하되 일부 요소를 추가·수정·삭제하여 최적화함. 그 결과 6개 차상위요소, 29개 하위요소, 52개 최하위요소, 36개 세부요소로 구성된 메타데이터(안)을 도출함

○ (향후 계획 제안) 도출된 장기보존 메타데이터(안)을 실제 데이터세트에 시범 적용하여 실효성을 검증할 필요가 있음

라. 장기보존패키지의 '기술정보(Descriptive Information)' 설계

○ (주요연구결과) 장기보존패키지의 기술정보는 OAIS 참조모형 정보패키지의 구성요소로서, 패키지 및 구성요소에 대한 이용자의 접근(검색, 열람, 분석, 청구)을 지원함. 이를 전 세계적으로 널리 활용되는 메타데이터 표준인 Dublin Core와 매핑함으로써 장기보존 메타데이터 요소 간 의미적 일관성과 상호운용성을 확보함

○ (향후 계획 제안) 향후 연구에서는 선택사항으로 제시된 EAD(Encoded Archival Description)의 정의와 적용 방안에 대한 구체적 설계를 수행함으로써 기술정보 설계의 완성도와 활용성을 높이고자 함

마. 장기보존패키지(NEO3) 생성 및 검증 방안 설계

○ (주요연구결과) 장기보존패키지(NEO3) 생성 및 검증 방안을 ① 메타데이터와 원문·붙임 확보, ② 원문·보존포맷 등 폴더 배치, ③ 메타데이터 확정, ④ 전자서명 적용, ⑤ 패키징 완료의 5단계로 설계함

○ (향후 계획 제안) 새로운 장기보존 메타데이터 정의, 구조 설계, 보존 대상 결정 등 여러 요소가 복합적으로 얽혀 있으며, 연구나 기술적인 문제와는 달리, 정책적 판단이 필요한 경우도 있음. 원기관 및 주관기관의 공동 논의와 연구를 통해 생성 및 검증 방안을 설계할 수 있었으나 추후 기록관리시스템에 적용할 경우 정책적, 기술적 관점에서 다시 한번 검토할 것을 제안함

바. 장기보존패키지(NEO3) 변경이력(Change History) 관리방안 설계

○ (주요연구결과) 본 연구는 장기보존패키지(NEO3)의 변경이력 관리 방안을 제시함. 최신본은 루트에 유지하고, 변경 전 파일만 이력으로 보관하며, 전자서명과 패키징 정보를 활용해 무결성을 검증하고 안정적으로 복원할 수 있도록 설계함

○ (향후 계획 제안) 생성되거나 변환된 모든 장기보존패키지(NEO3)는 반드시 보존되어야 하고 무결성 검증이 가능해야 한다는 데이터 무결성(data integrity) 원칙과, 대량·대용량 기록물 구성요소를 가진 NEO3에서 중복 저장을 최소화해 시스템 효율을 높인다는 중복 최소화(redundancy minimization) 원칙을 전제로 함. 이를 바탕으로 최초 탐색 구성요소 복원 방식을 도출함으로써 두 가지 원칙을 동시에 충족하는 방안은 제안함

사. 장기보존패키지(NEO3) 변경이력(Change History) 복원방안 설계

○ (주요연구결과) 최초 탐색 구성요소 복원 방식을 도출하여 모든 버전 별로 기록물을 완전하게 복원할 수 있었음

○ (향후 계획 제안) 장기보존패키지 전체를 백업하는 문서 중심의 NEO2 방식은 대용량·대량 기록물에는 적합하지 않으므로, 이에 대응할 수 있는 새로운 복원 방안이 필요함. 최초 탐색 구성요소 복원 방식 도출하여 대용량·대량의 기록물에도 적합한 복원방안을 제안함

4.7 데이터세트 유형 보존포맷 및 장기보존패키지 생성을 위한 테스트베드 구축 및 실데이터 기반 검증

가. 실데이터 기반 데이터베이스 생성 및 테스트베드 환경 구축

○ (주요연구결과) 검증 대상 정보시스템 '오아시스 3.0'의 '개인인사기본' 메뉴를 기준으로 MySQL 기반의 관계형 데이터베이스를 생성함. 총 16개 테이블을 설계하고 493건의 테이블 데이터와 비구조화 객체 데이터(Trigger, Event 등)를 추가하여 보존포맷 생성의 테스트베드로 활용함

○ (향후 계획 제안) 데이터베이스 내 비구조화 객체 데이터(Trigger, Event 등)를 XML 파일로 변환 및 추출하고, 이를 보존포맷 재현 도구 설계 과정과 연계하고자 함

나. 보존포맷 생성 및 검증

○ (주요연구결과) DB형 데이터세트의 보존포맷 생성 절차에 따라 메타데이터(metadata.xml), 실제 데이터(content/), 첨부파일(attachment/), 재현파일(representation/), 무결성정보(manifest.xml)를 실제 구성하였으며, 특히 blob 데이터의 Base64 변환 및 외부 첨부파일의 경로 이관 방식을 명확히 정의함. 오라클, Myslq, MSSQL, 티베로 에 대한 보존포맷 생성 과정 설명함

○ (향후 계획 제안) 보다 다양한 DBMS 환경에서도 호환 가능한 범용 보존포맷 생성·검증 체계를 구축할 것을 제안함

다. 보존포맷 재현 도구(XSLT/XML) 생성 및 웹기반 기록물 재현

○ (주요연구결과) 웹기록물의 재현을 위해 XML, XSLT, HTML/CSS를 연계한 단계별 재현 프로세스를 확립하고, 웹서버 및 프로그램 기반의 재현 방식을 비교·검증하였으며, 이를 NEO3 장기보존 패키지 구조에 통합하여 메타데이터, 원문, 보존포맷, 전자서명, 이력관리 등을 포함한 검증 가능한 보존체계를 구현함

○ (향후 계획 제안) 보존포맷의 재현도구는 XSLT/XML뿐만 아니라 다른 형태로도 구현될 수 있으며, 다양한 재현도구를 활용해 웹 기반 기록물을 재현하는 방안을 마련할 필요가 있음

라. 장기보존패키지 생성 및 검증

○ (주요연구결과) NEO3 장기보존패키지의 검증을 위해 행정정보시스템 분석부터 폴더 구조 배치, 전자서명 적용, 이력관리, 패키징에 이르는 전 과정을 테스트베드로 구현하였으며, 실데이터 기반 검증을 통해 메타데이터의 전자서명값 일치 여부로 진본성과 무결성을 확인할 수 있음을 입증함

○ (향후 계획 제안) NEO3 패키지 자동 생성 및 검증 절차를 표준화하여, 장기보존패키지의 실무 적용성과 상호운용성을 강화할 필요가 있음

제5장 총괄연구개발과제의 연구성과

5.1 활용성과

총괄과제명	오픈 포맷을 활용한 행정정보 데이터세트 보존 방안 연구
총괄과제책임자	김병동 / ㈜그리너리 / 컴퓨터공학

가. 연구논문

번호	논문제목	저자명	저널명	집(권)	페이지	Impact factor	국내/국외	SCI여부
1	필수보존속성에 기반한 기록물 디지털화의 제도적 관리 체계 연구(11월 게재예정)	김누리, 양동민	한국기록관리학회지	25(4)			국내	
2								

나. 학술발표

번호	발표제목	발표형태	발표자	학회명	연월일	발표지	국내/국제
1							
2							

다. 지적재산권

번호	출원/등록	특허명	출원(등록)인	출원(등록)국	출원(등록)번호	IPC분류
1						
2						

라. 정책활용

· 국가기록원의 데이터세트 기록관리의 장기보존전략 및 정책에 활용
· 국가기록원 장기보존패키지(NEO3) 표준화 확장에 활용
· 영구보존으로 책정된 데이터세트 기록물의 장기보존전략에 활용

마. 타연구/차기연구에 활용

· 시청각기록물, 웹기록물 등 다른 유형의 전자기록물에도 범위를 확장하여 연구

바. 언론홍보 및 대국민교육 (해당 없음)

사. 기타 (해당 없음)

5.2 활용계획

○ 추후 표준화 추진 : 국가기록원 공공표준은 표준화작업반을 통해 재개정되므로, 내년 표준화작업반 참여를 추진하고, 표준화작업반 위원 활동을 통해서 장기보존패키지(NEO3) 공공표준 개정을 수행하고자 함

○ 공공표준을 개정을 통해 공공기관에서 생산되는 데이터세트 중에서 30년 이상으로 보존기간이 책정된 데이터세트 유형의 기록물에 적합한 장기보존전략에 활용

제6장 기타 주요 연구변경사항

○ 강지수 연구원 취업으로 인해 2025년 7월 1일부터 과제 미참여

○ 김 송 연구원 개인사정으로 인해 2025년 9월 1일부터 과제 미참여

○ 진엽엽 연구원 2025년 9월 1일부터 NEO3/보존포맷 설계 업무에 추가 인력 배치

제7장 참고문헌

【국내】

김나현 (2024). 한국 디지털 문화유산(Digital Heritage) 관리 방안: 유럽 유로피아나(Europeana)와 한국 e뮤지엄 사례를 중심으로. 디지털콘텐츠학회논문지, 25(8), 2319-2330. https://doi.org/10.9728/dcs.2024.25.8.2319

이정은, 김지혜, 왕호성, 양동민 (2022). 행정정보 데이터세트의 관리기준표 개선방안 연구. 한국기록관리학회지, 22(1). 177-200

이현실, 한성국, 전양승 (2005). MARC의 개념 모델링 연구. 한국도서관·정보학회지, 36(3), 275-289.

최광훈, 양동민 (2024). 국가 정보시스템의 행정정보 데이터세트의 기록관리를 위한 기록속성 기준 수립 방안 연구. 한국차세대컴퓨팅학회 논문지, 20(1), 30-49.

【국외】

Amerio, S., Behari, S., Boyd, J., Brochmann, M., Culbertson, R., Diesburg, M., ... & White, S. (2017). Data preservation at the Fermilab Tevatron. Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment, 851, 1-4.

Arbey, A., Stark, G., Diaconu, C., Schwickerath, U., Duellmann, D., Lange, C., ... & Ebert, M. (2025). arXiv: Data Preservation in High Energy Physics (No. DPHEP-2025-01).

Basaglia, T., Bellis, M., Blomer, J., Boyd, J., Bozzi, C., Britzger, D., ... & DPHEP Collaboration. (2023). Data preservation in high energy physics. The European Physical Journal C, 83(9), 795.

DCMI Usage Board. (2007). Dublin Core Metadata Element Set, Version 1.1 (RFC 5013). Internet Engineering Task Force. https://doi.org/10.17487/RFC5013

Ebert, M., Roney, M., & Sobie, R. The BABAR Long Term Data Preservation and Computing Infrastructure.

Ferreira, B., Faria, L., Ramalho, J. C., & Ferreira, M. (2024). Database Preservation Toolkit: A relational database conversion and normalization tool. THE journal 4, 1(1).

Lindley, A. (2013). Database preservation evaluation report - SIARD vs. CHRONOS. In iPRES 2013.

Marstein, K. E., Grytnes, J. A., & Lewis, R. J. (2024). ECKOchain: A FAIR blockchain based database for long term ecological data. Methods in Ecology and Evolution.

Rahman, A. U., Muzammal, M., David, G., & Ribeiro, C. (2015). Database Preservation: The DBPreserve Approach. International Journal of Advanced Computer Science and Applications,

6(12), 255-266.

Sharma, C., & Vaid, A. (2024). Leveraging SAP Information Lifecycle Management (ILM): Latest Insights and Applications. Zenodo, 5(6), 167-173.

【인터넷자료】

엑스소프트. (n.d.). 문서 아카이브 솔루션. Retrieved from <https://exsoft.cafe24.com/page/mnu02/0208.php>, 2025.06.21., 확인

CHRONOS. (n.d.). CHRONOS Official Website. Retrieved from <https://csp-chronos.com/>, 2025.06.22., 확인

CERN. (n.d.). CERN Open Data Portal. Retrieved from <https://opendata.cern.ch/>, 2025.06.29., 확인

CERN. (n.d.). CERN Figshare Portal. Retrieved from <https://figshare.com/>, 2025.06.30., 확인

DataONE. (n.d.). DataONE API Documentation. Retrieved from <https://dataoneorg.github.io/api-documentation/>, 2025.06.30., 확인

CERN. (2021). FNAL DPHEP June 2021 Presentation. Retrieved from <https://indico.cern.ch/event/1043155/contributions/4383095/attachments/2269882/3854730/FNAL%20DPHEP%20June%202021.pdf>, 2025.06.30., 확인

ETH Zurich. (n.d.). ETH Data Archive. Retrieved from <https://library.ethz.ch/en/archiving-and-digitising/archiving/digital-long-term-preservation/eth-data-archive.html>, 2025.06.23., 확인

ETH Zurich. (n.d.). File formats for archiving. Retrieved from <https://unlimited.ethz.ch/spaces/DD/pages/194127879/File+formats+for+archiving>, 2025.06.28., 확인

Europeana. (2023 September 11). EDM Documentation. Retrieved from <https://pro.europeana.eu/page/edm-documentation>, 2025.06.23., 확인

HEPData. (n.d.). HEPData Official Website. Retrieved from <https://www.hepdata.net/>, 2025.06.23., 확인

LOC. (2022 February 2). MARCXML Official Website. Retrieved from <https://www.loc.gov/standards/marcxml>, 2025.06.23., 확인

National Archives and Records Administration. (2023 May 4). AC 31.2023: Release of regulations with digitization standards for permanent records. Retrieved from <https://www.archives.gov/records-mgmt/memos/ac-31-2023>, 2025.08.14., 확인

National Archives and Records Administration. (n.d.). GRS 4.5: Digitizing records. Retrieved from <https://www.archives.gov/files/records-mgmt/grs/grs04-5.pdf>, 2025.08.14., 확인

OpenText. (n.d.). Information Archive. Retrieved from

<https://www.opentext.com/ko-kr/products/information-archive>, 2025.06.21., 확인

OpenText. (2010 July). Archiving and Document Access for SAP Solutions Whitepaper. Retrieved from <https://www.opentext.com/file_source/OpenText/en_US/PDF/OpenText%20Archiving%20and%20Document%20Access%20for%20SAP%20Solutions%20Whitepaper.pdf>, 2025.06.21., 확인

OpenText. (n.d.). OpenText InfoArchive. Retrieved from <https://www.opentext.com/file_source/OpenText/en_US/PDF/opentext-po-infoarchive.pdf>, 2025.06.21., 확인

Public Record Office Victoria. (n.d.). Digitisation. Retrieved from <https://prov.vic.gov.au/recordkeeping-government/a-z-topics/digitisation>, 2025.08.14., 확인

Public Record Office Victoria. (n.d.). PROS 10/13 G2: Implementing disposal program. Retrieved from <https://prov.vic.gov.au/recordkeeping-government/document-library/pros-1013-g2-implementing-disposal-program>, 2025.08.14., 확인

Public Record Office Victoria. (n.d.). PROS 19/07: Converted or digitised records. Retrieved from <https://prov.vic.gov.au/recordkeeping-government/document-library/pros-1907-converted-or-digitised-records>, 2025.08.14., 확인

Public Record Office Victoria. (n.d.). PROS 22/04: Disposal standard. Retrieved from <https://prov.vic.gov.au/recordkeeping-government/document-library/pros-2204-disposal-standard>, 2025.08.14., 확인

Public Record Office Victoria. (n.d.). PROS 25/02 S1: Digitisation specification. Retrieved from <https://prov.vic.gov.au/recordkeeping-government/document-library/pros-2502-s1-digitisation-specification>, 2025.08.14., 확인

SAP. (n.d.). Information Lifecycle Management (ILM) Official Website. Retrieved from <https://www.sap.com/products/technology-platform/information-lifecycle-management.html>, 2025.06.28., 확인

SOLIXCloud. (n.d.). Database Archiving. Retrieved from <https://www.solix.com/products/database-archiving/>, 2025.06.28., 확인

Society of American Archivists. (1999). EAD application guidelines for version 1.0. Retrieved from <https://www.loc.gov/ead/ag/agappb.html>, 2025.07.16., 확인

Technical Subcommittee for Encoded Archival Standards. (2023). Encoded archival description tag library: Version EAD3 1.1.2 (2023 ed.). Society of American Archivists. Retrieved from <https://www.loc.gov/ead/EAD3taglib/>, 2025.07.16., 확인

U.S. General Services Administration. (2021 March). NARA FERMI use cases for disposal. Retrieved from <https://ussm.gsa.gov/assets/files/ERM/NARA%20FERMI%20Use%20Cases%20for%20Disposal

%20March%202021.pdf>, 2025.08.14., 확인

U.S. General Services Administration. (2021 March). NARA FERMI use cases for transfer. Retrieved from <https://ussm.gsa.gov/assets/files/ERM/NARA%20FERMI%20Use%20Cases%20for%20Transfer%20March%202021.pdf>, 2025.08.14., 확인

U.S. Government Publishing Office. (2023). Code of Federal Regulations, Title 36, Part 1236 - Electronic Records Management. Retrieved from <https://www.ecfr.gov/current/title-36/part-1236>, 2025.06.19., 확인

제8장 첨부서류

[첨부01] 전자기록물 장기보존 패키지 기술 규격 표준(안)
[첨부02] 2025년 11월 게재 예정 KCI등재지 논문
[첨부03] 장기보존패키지 검증 소프트웨어 사용설명서
[첨부04] 데이터세트(DB형) 유형 보존포맷 규격 표준(안)

제9장 별첨자료

- 장기보존패키지 검증 소프트웨어

오픈 포맷을 활용한 행정정보 데이터세트
보존 방안 연구

첨 부

첨부 목차

오픈 포맷을 활용한 행정정보 데이터세트
보존 방안 연구

[첨부01] 전자기록물 장기보존 패키지 기술 규격 표준(안)

기록관리 공공표준

NAK 31-2:20xx(vx.x)

National Archives Standard

전자기록물 장기보존패키지 기술규격-제2부: 디렉토리로 구조화된 방식(NEO3)(vx.x)

Technical Specification for Long-Term Preservation Package
Part 2: Directory Structured Format(NEO3)

Version x.0

국가기록원
National Archives of Korea

20xx년 xx월 xx일 제정

○ 제 정 자 : 행정안전부 국가기록원장

○ 제 정 일 : 20xx년 xx월 xx일(국가기록원 고시 제20xx-xx호)

○ 심 의 : 국가기록관리위원회, 기록·정보정책전문위원회

○ 원안작성 :

○ 검 토 :

•

○ 관 리 :

• 국가기록원 정책기획과

(1) 이 표준의 열람은 홈페이지를 이용하시고, 의견 또는 질문은 아래 전화로 연락 주십시오.

• 표준열람 : 국가기록원(http://www.archives.go.kr)→기록관리업무→기록관리표준→표준화현황

• 행정안전부 국가기록원 기록정책부 디지털기록혁신과(042-481-6331)
기록정책부 정책기획과(042-481-6231)

목 차

머리말

이 표준은 전자기록물의 진본성·무결성을 유지하고, 장기간 안전하게 보존할 수 있도록 전자기록물을 캡슐화하여 관리하는 장기보존패키지 기술규격을 제시하기 위해 제정되었다.

이 표준은 「공공기록물 관리에 관한 법률」 시행령 제36조(기록관 및 특수기록관의 전자기록물 보존), 동 시행령 제40조(기록관 및 특수기록관의 소관 기록물 이관)에서 중앙기록물관리기관의 장이 정하도록 한 사항을 규정한다.

이 표준에서 다루고 있는 '장기보존패키지'는 「공공기록물 관리에 관한 법률」 시행령 제36조(기록관 및 특수기록관의 전자기록물 보존)의 '장기보존패키지'를 의미하는 것이다. 그리고 장기보존패키지를 생성하는 방식별로 표준을 구분하기 위하여 NAK 31「전자기록물 장기보존포맷 기술규격」을 NAK 31-1「전자기록물 장기보존패키지 기술규격-제1부: XML로 포맷화된 방식(NEO2)」과 NAK 31-2「전자기록물 장기보존패키지 기술규격-제2부: 디렉토리로 구조화된 방식(NEO3)」으로 분리하였다.

2022년 1차 개정에서는 「공공기록물 관리에 관한 법률」 시행령 개정에 따라 '장기보존포맷', '문서보존포맷'의 명칭이 각각 '장기보존패키지', '보존포맷'으로 용어 변경된 사항을 반영하였다.

이 표준은 디지털객체의 장기보존패키지 생성 시 원문 및 보존포맷로 구성되는 콘텐트 정보, 장기보존 메타데이터, 패키징 정보, 진본확인 정보 등을 위해 디렉토리 구조로 구성하고 논리적인 연관관계와 구성 정보를 대표적인 마크업언어(Markup Language)인 XML로 기술한다. 그래서 디지털 객체의 특정 기술에 대한 종속성을 제거하고, 정보에 대한 접근성을 개선하였으며, 다양한 디지털 기록물의 유형에 따른 메타데이터 변경 등을 고려하여 확장성 있고 유연한 구조를 적용하고자 하였다.

표준의 구성은 다음과 같다. 제1절부터 제3절까지는 표준의 적용범위, 적용근거 및 용어를 정의하였다. 제4절에서는 장기보존패키지의 개요 및 두 가지 생성방식에 대해 기술하였고, 제5절에서는 장기보존패키지 생성방식 중 제2부 표준에서 규정하는 디렉토리로 구조화된 방식의 장기보존패키지(NEO3)에

대한 구조 및 패키지 정보, 콘텐트 정보, 메타데이터, 진본확인 기술, 장기보존패키지 변경이력 관리, 도큐멘테이션 등에 대해서 기술하였다. 제6절에서는 기록물 유형에 따라 장기보존패키지 생성 과정을 구분하여 기술하였고, 기록물 유형별 장기보존패키지 생성 과정과 공통 과정을 구분하였다. 제7절에서는 장기보존패키지 변경이력 관리에 대한 설명을 구체적으로 기술하였고, 변경이력 관리 방안과 복원 방안 과정을 기술하였다. 부속서에는 NEO3 장기보존패키지 구성 사례를 수록하였다.

이 표준은 기록·정보정책전문위원회 및 국가기록관리위원회 심의를 거쳐 제정되었으며, 국가기록원이 유지・관리한다. 이 표준은 관련 법령의 개정, 관계 기관 및 이해당사자의 요청 등 개정사유가 발생할 경우 그 필요성 및 타당성을 검토 후 개정안을 마련하고 의견수렴 및 심의 절차를 거쳐 개정한다.

전자기록물 장기보존패키지 -
제2부: 디렉토리로 구조화된 방식(NEO3)

1 적용범위

이 표준은 디지털객체에 대한 논리적 또는 물리적 캡슐화 규격을 규정하는 동시에 진본성, 무결성, 이용가능성을 유지하기 위한 디지털기록물에 대한 장기보존패키지에 대한 기술규격을 기술하는 것으로, 「공공기록물 관리에 관한 법률」에 명시된 공공기관 및 기록물관리기관이 전자기록물을 장기간 보존해야 하는 경우 적용할 수 있다.

다만, 영구기록물관리기관, 동법 시행령 제3조 각 호에 해당하는 공공기관 등 전자기록물을 장기간 보존하는 기록물관리기관(이하 '영구기록물관리기관 등'으로 칭함)은 소장 기록물의 특성, 기관의 환경 등에 따라 전자기록물의 장기보존패키지 방식을 달리 정할 수 있다.

비고 1 장기보존패키지'는 ISO 14721 OAIS(Open Archival Information System) 참조모델의 정보 패키지(Information Package)를 참고한다.

비고 2 이 표준에서 제시되는 장기보존패키지는 ISO 14721 OAIS(Open Archival Information System) 참조모형에서 말하는 기록관에서 영구기록물관리기관으로 전자기록물을 이관하는 포맷(SIP, Submission Information Package), 이용자에게 전자기록물을 제공하는 포맷(DIP, Dissemination Information Package) 등으로도 활용될 수 있다.

2 적용근거

2.1 법적 근거

이 표준의 구체적인 법적 근거는 다음과 같다.

- 「공공기록물 관리에 관한 법률」 제20조(전자기록물의 관리)
- 「공공기록물 관리에 관한 법률」 시행령 제36조(기록관 및 특수기록관의 전자기록물 보존)
- 「공공기록물 관리에 관한 법률」 시행령 제40조(기록관 및 특수기록관의 소관 기록물 이관)
- 「공공기록물 관리에 관한 법률」 시행령 제46조(영구기록물관리기관의 전자기록물 보존 및 관리)

2.2 인용표준

다음의 인용표준은 이 표준의 적용을 위해 필수적이다. 발행연도가 표기된 인용표준은 인용된 판만을 적용한다. 발행연도가 표기되지 않은 인용표준은 최신판(모든 개정내용을 포함)을 적용한다.

- ISO 15489-1:2016, Information and documentation - Records management - Part 1 : Concepts and principles
- ISO 23081-1:2017, Information and documentation - Records management processes - Metadata for records - Part 1 : Principles
- ISO 14721:2012, Space data and information transfer systems - Open archival information system(OAIS) - Reference model
- NAK 8 2016(v2.1) 기록관리 메타데이터 표준

2.3 다른 표준과의 연계

이 표준의 적용을 위해 필요하거나 직접적으로 연관이 있는 표준은 다음과 같다. 발행연도가 표기되지 않은 표준은 최신판(모든 개정내용을 포함)을 적용한다.

- KS X 6030 확장가능한 마크업 언어 (XML) - Extensible Markup Language (XML) 1.0
- KS X 6033 확장가능한 마크업 언어 전자서명 구문과 처리 - XML Signature Syntax and Processing
- ISO ISO 26324:2012 Information and documentation − Digital object

identifier system
- RFC3280/RFC5280(Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List Profile)
- NAK 6:2015(v1.5) 기록관리시스템 기능 요건
- NAK 7:2015(v1.5) 영구기록관리시스템 기능 요건
- NAK 19-1:2012(v1.0) 전자기록생산시스템 기록관리 기능요건
- NAK 29-1:2017(v1.4) 기록관리시스템 데이터연계 기술규격-제1부: 업무관리시스템과의 연계
- NAK 29-2:2022(v1.5) 기록관리시스템 데이터연계 기술규격-제2부: 중앙영구기록관리시스템과의 연계
- NAK 32-1:2022(v1.2) 전자기록물 전자서명 인증서 장기검증 기술규격(v1.1)
- 행정전자서명 인증업무지침(행정안전부고시 제2019-49호)
- NAK 35:2020(v1.0) 행정정보 데이터세트 기록관리 기준-행정정보 데이터세트 기록관리 기준-관리기준표 작성 및 이관규격

3 용어정의

이 표준의 목적을 위해 다음의 용어와 정의를 적용한다.

3.1 기록관리 메타데이터(Metadata for managing records)

시간과 공간을 초월하여 기록의 생산, 관리와 이용이 가능하도록 하는 구조화된 혹은 반구조화된 정보

[KS X ISO 23081-2, 3.7 기록관리 메타데이터, ISO 15489-1, 3.12 metadata for records]

3.2 디지털 객체 식별자 시스템(DOI system)

전자기록물 여부와 관계없이 모든 유형의 객체에 대해 항구적이고 고유하게 식별할 수 있는 체계. 디지털 객체 식별자(DOI)는 Digital Object Identifier의 약어이며 ISO 26324:2012으로 표준화되었다. DOI는 인터넷 상에서 동작하도록 설계되었으며 객체의 물리적 위치(IP 주소, 디렉토리 경로 등)가 변경되어도 DOI명은 변하지 않는다. DOI시스템을 통해 해당 DOI명과 관련된 다양한 정보(URL, 이메일 주소, 다른 식별자 등)를 통해 객체에 접근할 수 있다.

3.3 스키마(Schema)
일반적으로 메타데이터의 사용과 관리, 특히 의미론, 구문론, 설정값(강제 수준)에 관한 규칙 수립을 통해 메타데이터 요소 간 관계를 보여주는 논리 도표

3.4 에뮬레이션(Emulation)
원래의 소프트웨어 기능이 현재의 컴퓨터 환경에서 재생산될 수 있도록 하는 것으로 전자기록물의 원래 운영환경을 재생산하는 소프트웨어를 사용하게 하는 보존방법

3.5 인증(Certification)
전자서명키가 가입자에게 유일하게 속한다는 사실을 확인하고 이를 증명하는 행위

3.6 인증기관(Certificate Authority)
사용자 및 다른 인증기관들에게 공개키 인증서를 발급하는 기관. 인증서 폐기 목록을 주기적으로 발행하며, 디렉토리에 인증서와 인증서 폐기 목록을 게시한다.

3.7 인증서(Certificate)
공개키와 사용자의 관계를 연결하여 주는 전자 정보. 인증기관에서 발급한다.

3.8 인코딩(Encoding)
기계적 처리를 위해 고안된 구문 또는 신호를 특정한 부호들의 나열로 그 형태를 바꾸는 것
[정보통신용어사전, KS X ISO 23081 참조하여 개작]

3.9 장기보존(Long-term Preservation)
오랜 시간이 경과된 후에도 전자기록물에 접근할 수 있고 진본의 상태를 유지하여 증거로서 인정받을 수 있도록 하는 보존행위

3.10 장기보존패키지(Long-term preservation package)
장기간 전자기록물의 진본성 및 무결성을 유지하기 위하여 전자기록물의 원

문, 보존포맷, 메타데이터, 진본확인 정보를 하나의 논리적 아니면 실제적으로 캡슐화한 객체

3.11 전자서명(Digital signature)

서명자를 확인하고 서명자가 해당 전자문서에 서명을 하였음을 나타내는데 이용하기 위하여 해당 전자문서에 첨부되거나 논리적으로 결합된 전자적 형태의 정보. 즉, 전자서명은 서명자 확인, 부인 방지, 위변조 확인을 위해 생성될 수 있다. 전달된 메시지나 문서의 원래 내용이 변조되지 않았다는 것을 보증하기 위해 사용될 수 있다.
[「전자서명법」제2조제2항 참조하여 개작]

3.12 전자서명 알고리즘(Digital signature algorithm)

전자서명에 사용되는 알고리즘으로 해시 값을 통해 전자서명 방식을 결정하는 것. SHA1WithKCDSA 또는 SHA256WithRSA 등으로 표시한다.

3.13 진본확인 정보(Authentic identification information)

전자기록물의 진본성 및 무결성 유지를 위해 장기보존패키지에 포함되는 파일별 해시정보 등 전자서명 관련 정보로 GPKI 인증서, XML 전자서명 파일 등이 포함될 수 있다.

3.14 캡슐화(Encapsulation)

전자기록물에 메타데이터를 디지털 객체와 함께 하나로 묶거나 디지털 객체에 포함시키는 방법.

3.15 컨테이너(Container)

상이한 데이터 구성요소들과 메타데이터를 하나의 객체로 묶을 수 있게 하는 방법. 그 방법을 기술하고 있는 파일 형식을 컨테이너 포맷(Container Format)이라 한다.

3.16 컴포넌트(Component)

기록물건을 구성하는 기록물의 최소단위. 일반문서 유형인 경우 종이문서, 전자문서의 본문, 첨부 데이터파일 등이 기록물건을 구성하는 컴포넌트가 되고, 다른 기록 유형인 경우 각 기록물건을 구성하는 고유의 컴포넌트 형식을 가진다.

3.17 해시함수(Hash function)

임의의 길이의 문자열을 고정된 길이의 이진 문자열로 매핑하여 주는 함수. 데이터를 자르고, 치환하거나 위치를 바꾸는 방법들로 결과를 만들어 내며, 이 결과를 해시 값(hash value)이라 한다. 해시 함수는 데이터의 무결성, 인증, 부인 방지 등에서 응용되는 중요한 함수 가운데 하나이다.
[TTA 정보통신용어사전]

3.18 행정전자서명(GPKI Digital Signature)

전자문서를 작성한 행정기관 또는 행정기관과 전자문서·행정정보를 유통하는 법인·단체에서 직접 업무를 담당하는 자의 신원과 전자문서의 변경 여부를 확인할 수 있는 정보

3.19 확장성 스타일시트 언어 변환(eXtensible Stylesheet Language Transformations, XSLT)

확장성 마크업 언어(XML) 문서를 다른 스타일의 XML 문서로 변경하거나 다른 문서 형식으로 변환하기 위해 개발된 언어. XSLT 기반으로 문서 변환 시 원본 문서는 변경되지 않고 변환 결과로 새로운 문서가 생성된다. HTML, 텍스트(txt), PDF, 포스트스크립트(PostScript), 이미지(PNG 등), 한글(HWP) 등의 형식으로 변환할 수 있다.
[TTA 정보통신용어사전]

4 장기보존패키지 개요

4.1 OAIS의 정보 패키지와 장기보존패키지의 관계

이 표준에서 규정하고 있는 장기보존패키지는 ISO 14721 OAIS 참조모형의 정보 패키지(Information Package)를 참고한다. 정보 패키지는 **그림 1**과 같이 콘텐트 정보(Content Information), 보존기술정보(Preservation Description Information), 패키징 정보(Packaging Information), 기술 정보(Descriptive Information)로 구성되어 있다. 콘텐트 정보는 실제 보존의 대상인 기록물을 의미하며, 원문과 보존포맷으로 구성될 수 있다. 보존기술정보는 콘텐트 정보의 보존과 설명에 필요한 메타데이터로서 본 표준에서 규정하는 있는 장

기보존패키지에서는 장기보존 메타데이터로 표시한다. 패키징 정보는 콘텐트 정보와 보존기술정보를 실제적으로 아니면 논리적으로 묶고, 식별하고, 관련 짓는 정보이다. 기술정보는 이용자가 해당 정보 패키지를 검색할 때 활용될 수 있는 정보로 전자기록물 목록 정보에 해당된다.

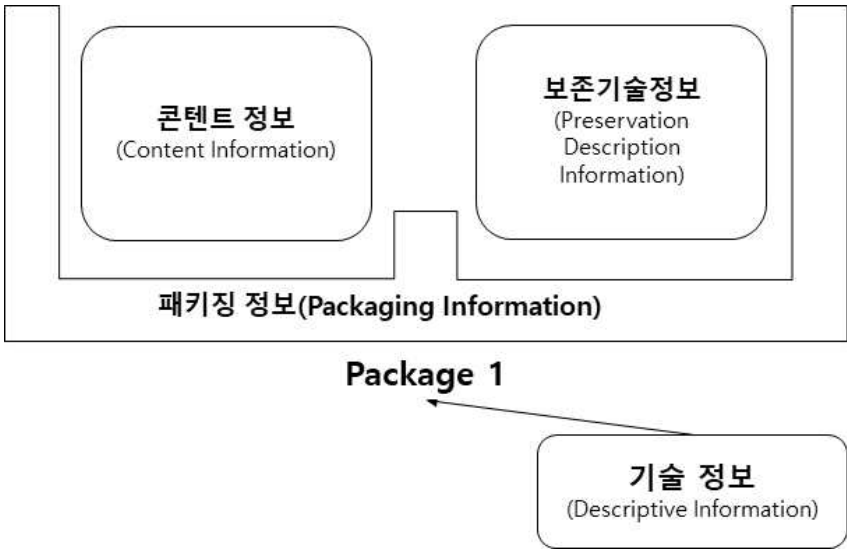


그림 1 - ISO 14721: OAIS 정보 패키지 개념 및 관계

장기보존패키지는 전자기록물의 진본성 · 무결성을 유지하고 장기간 안전하게 보존할 수 있도록 장기보존 메타데이터와 원문 및 보존포맷을 실제적 아니면 논리적으로 캡슐화하여 관리하는 개념적인 컨테이너이다. 즉, 장기보존패키지는 전자기록물 자체와 장기보존을 위한 보존포맷 및 메타데이터를 포함할 수 있도록 설계되었으므로 ISO 14721 OAIS 참조모델에서 제시하고 있는 정보 패키지(SIP, DIP, AIP) 중에서 AIP에 해당한다.

4.2 장기보존패키지 필요성

전자기록물은 그 특성상 변형, 훼손, 유실되기 쉬우므로 보다 안전하게 보존할 수 있도록 하는 기술적 기반을 필요로 한다.

즉, 오랜 기간이 경과해도 기록물이 생산된 당시에 가지고 있던 내용을 그대로 재현하여 접근할 수 있도록 보존되어야 하며, 업무 활동에 대한 증거와 업무에 대한 책임 소재를 분명하게 밝혀주는 법적 증거를 확보할 수 있도록 하여야 한다.

기록물이 가진 법적 증거는 기록물의 진본성과 무결성이 유지되어야만 확보될 수 있으므로 전자기록물에 대해 이러한 진본성과 무결성을 유지하면서 장기간 보존할 수 있게 하도록 하는 장기보존패키지의 적용이 필요하다.

4.3 장기보존패키지 고려사항

4.3.1 자체 충족성

기록물이 생산된 당시의 내용을 그대로 재현하여 읽고 이해할 수 있도록 하기 위하여 전자기록물은 시스템, 외부 데이터 등에 독립적이어야 한다.

시스템 의존적일 경우 시스템의 손실은 전자기록물의 손실이 되고, 외부데이터의 손실 역시 전자기록물의 손실이 되기 때문에 전자기록물은 자체적으로 기록물이 생산된 당시의 내용을 그대로 재현할 수 있는 충분한 능력을 가져야 한다.

4.3.2 자체 문서화

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 하여야 한다.

즉 기록물을 기술(記述)함과 동시에, 다른 기록물이나 기관 또는 조직과의 관계를 기술하고, 기록물의 지속적인 관리에 관련된 메타데이터를 포함하여야 한다. 메타데이터가 포함됨으로써 장기보존패키지가 독립적인 객체로서 기능할 수 있도록 한다.

4.3.3 진본성 및 무결성 유지

이관, 수집 등을 통해 영구기록물관리기관 등으로 제출(submission)된 전자기록물이 장기보존패키지로 변환되거나 저장ㆍ관리되는 동안 진본성을 유지할 수 있어야 한다. 이를 위해 전자서명 등의 기술을 활용한 진본확인 절차를

통해 검증할 수 있어야 한다.

전자기록물이 위조 또는 변조되지 않았음을 검증하여 기록물의 법적 증거를 확보하여야 한다. 장기보존패키지를 저장·관리하는 동안 무결성을 검증할 수 있어야 한다.

4.4 장기보존패키지 구성요소

장기보존패키지는 원문, 보존포맷, 장기보존 메타데이터, 진본확인 정보 등으로 구성된다. 이 구성 요소들은 장기보존패키지로 캡슐화되어야 하며, 캡슐화 방법은 국제표준 ISO 14721 OAIS 참조모형에 따르면, 논리적(logically) 아니면 실제적(actually) 방식으로 크게 구분한다. 캡슐화 방식에 대한 상세 유형은 **4.5 장기보존패키지 생성방식**을 참조한다.

장기보존패키지의 구성요소는 다음과 같다.

- **원문**: 생산자가 생산 또는 접수한 전자기록물 원본(진본)으로 진본성을 보장하기 위해 포함한다.

- **보존포맷**: 전자기록물 생산 당시의 애플리케이션이 없이도 해당문서의 내용과 외형을 그대로 재현한 포맷 또는 원본의 디지털객체 특성 정보(내용과 문맥정보, 유형별 특성 정보 등)를 원본과 동일한 수준으로 보존 가능한 포맷으로 시간과 기술변화에 상관없이 이용자가 기록물 내용에 접근할 수 있게 한다. 다만, 변환 가능한 보존포맷이 정해지지 않은 원문의 경우는 보존 대책을 마련할 때까지 원문만 포함할 수 있다.

- **장기보존 메타데이터**: 기록물의 생산부터 관리·보존에 이르는 전 과정을 기술(記述)한 정보로, 기록물 생애주기 전 기간에 걸쳐 진본성, 신뢰성, 무결성, 이용가능성을 유지하며, 기록물을 관리하고 보존·이해할 수 있도록 지원한다. 또한 패키지를 설명하는 패키징 정보 등 추가적인 메타데이터를 포함할 수 있다.

- **장기보존 메타데이터 스키마 정보**: 장기보존 메타데이터는 XML 문서로 저장된다. 메타데이터의 XML 구조는 시대가 흐르고 생산 시스템이 변화

함에 따라 지속적으로 변화된다. XML 스키마는 해당 패키지의 장기보존 메타데이터 유효성을 특정 시스템에 종속되지 않고 독립적으로 검증하기 위해 포함된다.

- **장기보존 메타데이터 스타일시트(StyleSheet)**: XML로 저장된 장기보존 메타데이터를 기록물의 생산·접수 시점의 화면과 유사하게 재현하기 위해 변환한 XSLT 스타일시트가 포함될 수 있다.

- **진본확인 정보**: 전자기록물의 진본성 및 무결성 유지를 위해 장기보존패키지에 포함되는 정보이다. 이 표준에서는 진본확인 기술 중 전자서명을 사용하며 관련 정보로 GPKI 인증서, XML전자서명 파일 등을 포함할 수 있다. 영구기록물관리기관 등이 전자서명 외의 다른 진본확인 기술을 적용하는 경우, 해당 기술에 따른 진본확인 정보를 관리할 수 있어야 한다.

비고 장기보존패키지 구성요소별 자세한 내용은 '**5 디렉토리로 구조화된 방식의 장기보존패키지(NEO3)**'를 참조한다.

4.5 장기보존패키지 생성방식

4.5.1 일반사항

장기보존 메타데이터와 전자기록물의 콘텐트 정보에 해당하는 원문 및 보존포맷은 국제표준 ISO 14721 OAIS 참조모형의 패키징 정보에 의해 실제적으로(actually) 아니면 논리적으로(logically) 묶고, 식별하고 관련지을 수 있다. 이에 따라, 장기보존패키지는 실제적 장기보존패키지와 논리적 장기보존패키지로 구분할 수 있다. 장기보존 메타데이터와 콘텐트 정보가 디지털 객체인 단일 파일로 생성되면 실제적 장기보존패키지이고, 장기보존 메타데이터와 콘텐트 정보가 각각 다른 파일로 존재하는 경우는 논리적 장기보존패키지라 한다.

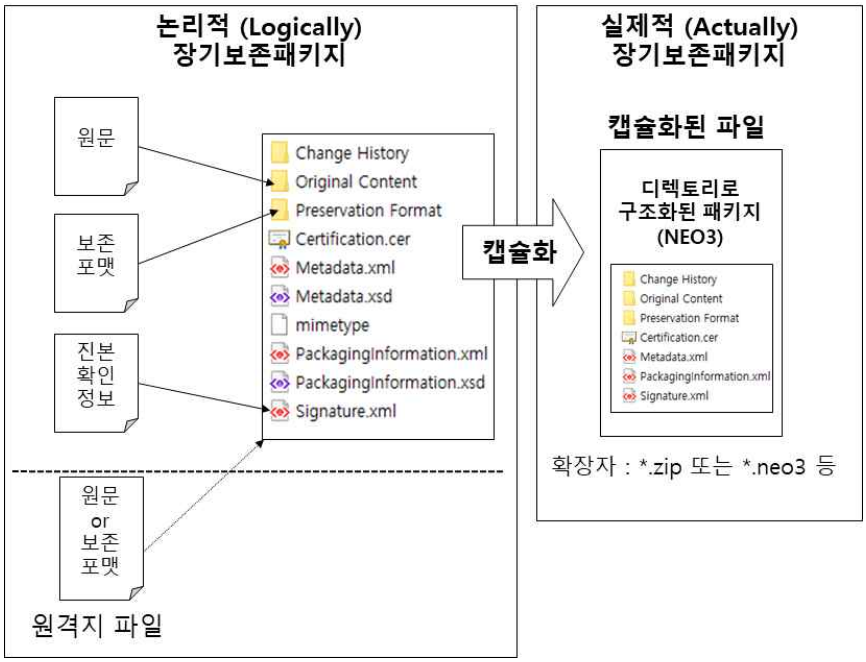


그림 2 - 논리적 · 실제적 장기보존패키지 사례

장기보존패키지는 **그림 2**와 같이 장기보존 메타데이터와 원문 및 보존포맷을 단일한 디지털 객체로 생성하는 방식의 실제적 장기보존패키지 또는 여러 디지털 객체들로 나뉘어 있는 것을 서로 연결하는 방식인 논리적 장기보존패키지가 모두 가능하다.

논리적 장기보존패키지는 구성요소인 장기보존 메타데이터, 원문 및 보존포맷 등을 패키징 정보에 포함된 식별자를 통해 서로 연결, 식별이 가능하며, 패키징 정보는 이를 위해 URL(Uniform Resource Locator), URN(Uniform Resource Name), URI(Uniform Resource Identifier) 등을 포함한다. 원문 파일이나 보존포맷이 동일한 물리적 스토리지에 존재하지 않거나, 진본확인 기술의 보안성 강화를 위해 전자서명 등 진본확인 정보가 저장된 파일만을 보안이 강화된 별도의 스토리지에 보존하는 등의 경우에 논리적 장기보존패키지 방식을 적용할 수 있다. 다만, 이 장기보존패키지 방식은 장기보존 메타데이터 또는 원문 및 보존포맷의 물리적인 위치가 변경되더라도 패키징 정보의 논리적인 연결 관계가 단절되지 않도록 장기간 지속적으로 확인할 수 있는 별도의 식별자 검증 체계가 필요하다.

논리적 장기보존패키지는 실제적으로 하나의 객체로 캡슐화하지 않지만 장기보존 메타데이터, 패키징 정보, 원문, 보존포맷, 진본확인 정보를 논리적으

로 캡슐화함으로써 장기보존패키지의 구성요소가 서로 물리적으로 떨어져 있는 것을 허용한다. 앞서 언급한 바와 같이 이 패키지 방식을 적용하기 위해서는 원격지에 존재하는 디지털 객체의 위치정보 및 구분 정보를 관리하기 위한 디지털 객체의 식별자 관리 시스템이 마련되어야 한다. 이를 위해 디지털 객체 식별자(DOI) 시스템과 같은 방식이 사용될 수 있으며 이를 통해 각각의 요소를 영구적으로 연계하고 관리할 수 있다.

이 유형의 캡슐화 방식은 정보시스템, 매체 등 다양한 형태로 분산 저장되어 유실된 기록물에 대한 복구가 더 어려울 수 있다. 그러나 대용량의 전자기록물을 장기 보존하거나 비전자기록물과 전자기록물을 연계하는 경우, 특히 영구기록물관리기관 등에 이관된 대용량 멀티미디어 기록물의 캡슐화를 위해 대용량으로 이동, 복사 등 장기보존 처리 절차가 힘든 경우에는 불가피하게 논리적으로 캡슐화하는 방식으로 장기보존패키지를 생성할 수 있다.

실제적 장기보존패키지 방식은 현행 장기보존포맷인 NEO2(NAK Electronic Object 2)가 해당되며, 공공표준 'NAK 31-1:2022(v2.3) 전자기록물 장기보존패키지 기술규격 - 제1부: XML로 포맷화된 방식(NEO2)'에서 그 내용을 상세하게 규정하였다. 논리적 장기보존패키지 방식은 NEO3(NAK Electronic Object 3)가 해당되며, 공공표준 'NAK 31-2:2022(v1.1) 전자기록물 장기보존패키지 기술규격 - 제2부: 디렉토리로 구조화된 방식(NEO3)'에서 그 내용을 규정하였다. NEO3는 논리적 장기보존패키지로 설계되었지만 ZIP64 등으로 압축 또는 압축 없이 하나의 디지털 객체로 묶여서 보관되면 실제적 장기보존패키지가 된다.

NEO3에서는 기록물 생산·보존 기관에서 장기보존패키지를 생성할 때, 장기보존 메타데이터가 포함된 파일(Metadata.xml) 및 패키징 정보 파일(PackagingInformation.xml)이 **그림 2**의 사례처럼 최상위 디렉토리에 위치해야 하는 규칙을 제외하고는 기관별 상황에 따라 적용할 수 있다. 이를 위해 장기보존패키지 내 물리적으로 구성되는 디렉토리의 구조와 명칭은 별도 정의하지 않고 장기보존 메타데이터(Metadata.xml)의 패키징 정보(PackagingInformation.xml)의 내부 경로 정보로 확인할 수 있도록 한다. 디렉토리로 구조화된 방식의 장기보존패키지(NEO3)를 논리적 장기보존패키지로 구현한 사례는 **그림 3**에서 볼 수 있다.

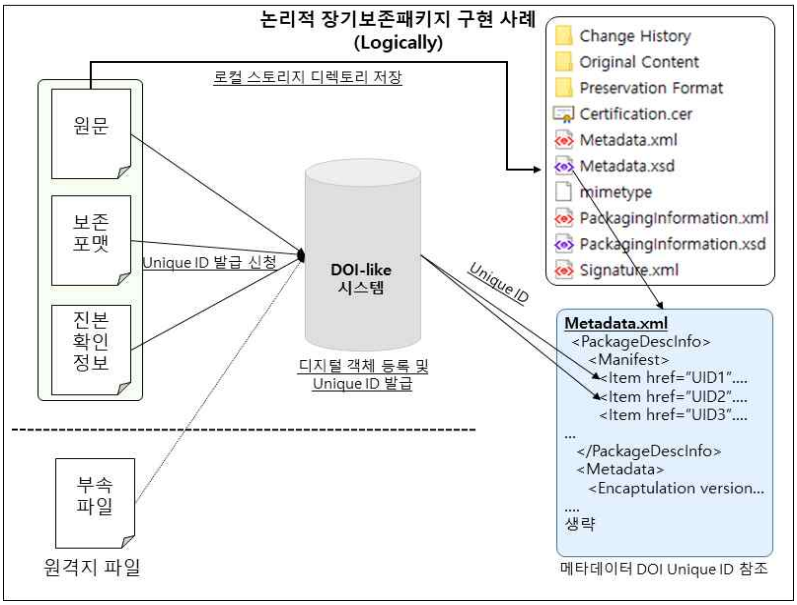


그림 3 - 논리적 장기보존패키지 구현 사례(NEO3)

XML로 포맷화된 방식의 장기보존패키지 또는 아카이브 포맷으로 압축된 디렉토리로 구조화된 방식의 장기보존패키지에 포함되는 컴포넌트 중 일부 항목의 위치를 다른 서버에 있는 저장소 등으로 지정하거나 비전자 문서를 연결하는 경우 실제적 장기보존패키지와 논리적 장기보존패키지의 혼합 방식이 된다.

장기보존패키지를 생성하는 방식은 실제적 장기보존패키지로 설계된 **4.5.2 XML로 포맷화된 방식**과 논리적 장기보존패키지로 설계된 **4.5.3 디렉토리로 구조화된 방식**으로 구분한다. 다만, 디렉토리로 구조화된 방식의 논리적 장기보존패키지가 ZIP64 등으로 압축 또는 압축 없이 하나의 디지털 객체로 묶여서 관리되는 경우에는 실제적 장기보존패키지라고 할 수 있다. 이때, 사용되는 압축 기술은 압축하는 파일의 크기의 제한, 파일의 수, 파일명에 대한 제한 등이 64비트 주소 체계가 지원되는 ZIP64 등 시장표준(De Facto) 기술을 활용하도록 한다.

4.5.2 XML로 포맷화된 방식

이 방식은 전자기록물 및 관련 메타데이터 정보를 단일 XML 파일에 모두 포함한다. 즉, 보존대상인 전자기록물을 하나의 객체처럼 캡슐화하는 방식이다. 다르게 말하면 XML로 포맷화된 방식이라고 할 수 있다. 원문과 보존포

맷이 바이너리(binary) 데이터라 하더라도 base64방식으로 인코딩함으로써 장기보존 메타데이터, 원문 및 보존포맷, 진본확인 정보 등을 텍스트 형식인 단일 XML 문서 내에 객체로 캡슐화한다. 이 방식을 채택한 해외 사례로는 호주 빅토리아 주립 기록보존소의 VEO2, 호주 국가기록원의 XENA 등이 있다.

이 방식은 공공표준 'NAK 31-1:2020(v2.2) 전자기록물 장기보존패키지 기술규격-제1부: XML로 포맷화된 방식(NEO2)(v2.2)'에서 기술규격을 상세하게 정하였다. 영구기록물관리기관 등은 XML로 포맷화된 방식을 채택하는 경우라 하더라도 기관의 환경 등에 따라 NEO2가 아닌 다른 장기보존패키지를 정의하고 적용할 수 있다.

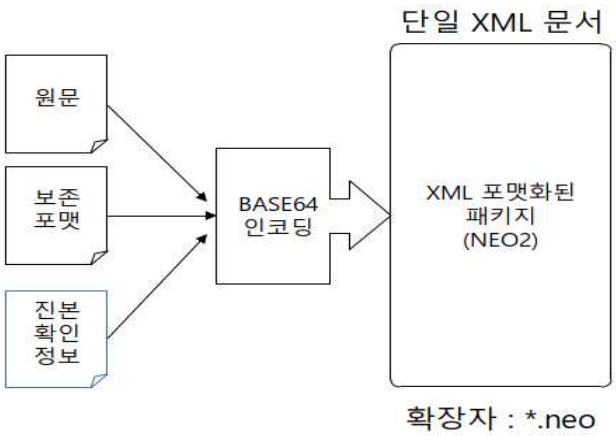


그림 4 - XML로 포맷화된 장기보존패키지

4.5.3 디렉토리로 구조화된 방식

이 유형은 보존대상인 전자기록물을 디렉토리로 구조화하여 캡슐화하는 방식으로 NEO2 패키지 방식과 달리 장기보존 메타데이터, 원문 및 보존포맷, 진본확인 정보를 각각 디렉토리와 하위 디렉토리 등에 구분하여 보관하는 캡슐화 방식이다. 디렉토리로 구조화된 패키지는 ZIP64 등으로 압축 또는 압축 없이 하나의 디지털 객체로 묶여서 보관될 수 있다. 이와 유사한 방식을 채택한 해외 사례로는 미국 의회 도서관의 BagIt, 호주 빅토리아 주립 기록보존소의 VEO3 등이 있다.

이 방식은 본 표준에 기술규격을 상세하게 정하였다. 영구기록물관리기관 등

은 디렉토리로 구조화된 방식을 채택하는 경우라 하더라도 기관의 환경 등에 따라 NEO3가 아닌 다른 캡슐화 방식을 정의하고 적용할 수 있다.

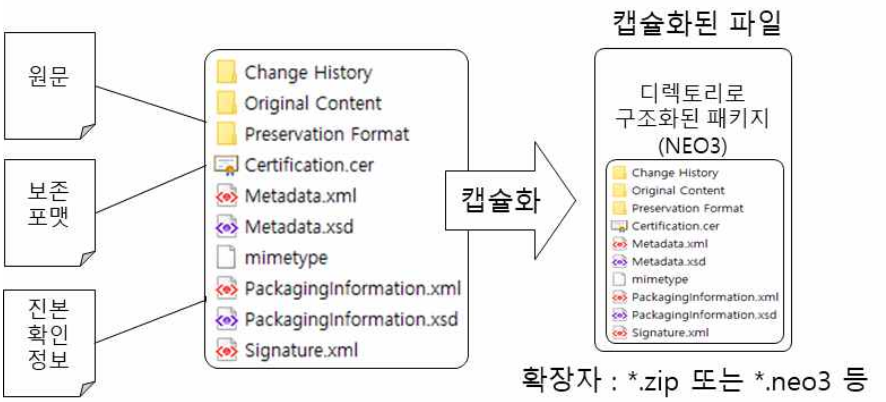


그림 5 - 디렉토리로 구조화된 장기보존패키지

5 디렉토리로 구조화된 방식의 장기보존패키지 구성요소 및 개요(NEO3)

5.1 디렉토리 구조

NEO3 장기보존패키지 방식은 개념적으로 기관이 정한 규칙에 따라 계층적 디렉토리 구조로 연관된 파일들을 분류·관리하며, 최상위 디렉토리를 기준으로 ZIP64 등으로 압축 또는 압축 없이 하나의 디지털 객체로 묶어서 보관하여 보존·활용할 수 있다.

장기보존패키지 내 포함되는 모든 파일들에 대한 논리적인 구성은 패키징 정보에 포함되어 장기보존패키지 내의 파일들에 대한 접근과 확인이 가능하다. 또한 기록물 이외 장기보존패키지에 추가적으로 포함되는 전자서명용 인증서나 이력정보 등 부속파일에 대한 파일별 해시 값과 파일 크기 정보를 패키징 정보에 포함한 경우 해당 파일의 무결성(네트워크 전송, 파일 복사 등으로 발생할 수 있는 오류)을 확인하기 위해 사용될 수 있다. 또한 패키징 정보에 포함되는 파일의 경로 정보는 URL, URI, URN 등의 정보로 원격지의 위치를 포함할 수 있어 실제적으로 캡슐화하지 않은 논리적 장기보존패키지로 활용할 수 있다. (예 : 변경이력정보나 전자서명 정보를 물리적으로 분리된 별도 서버에 두는 경우 등)

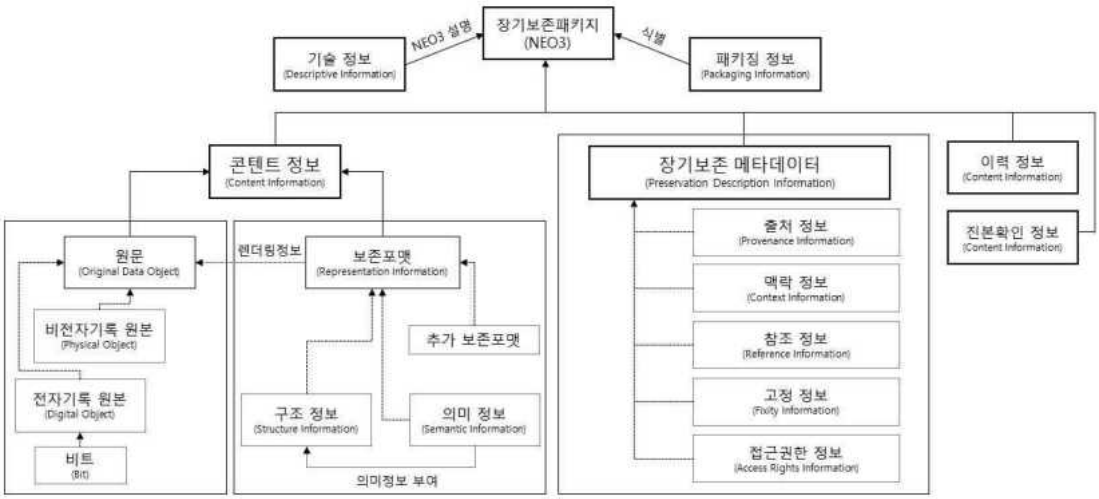


그림 6 - NEO3 개념적 구조

NEO3의 개념적인 구조는 **그림 6**과 같고, 그 구조는 ISO 14721 OAIS 참조 모형의 정보 패키지(Information Package)를 참고하여 설계되었다. OAIS 정보 패키지의 보존기술정보(Preservation Description Information)에 해당하는 구성요소의 명칭을 장기보존 메타데이터로 한 것을 제외하고는 모두 OAIS 정보 패키지의 구성요소와 명칭을 따른다. NEO3 장기보존패키지에는 해당 패키지에 대한 검색이나 추출 기능에 사용되기 위한 메타데이터를 포함할 수 있으며 이는 기술 정보이다. 원문 및 보존포맷 등 원본 기록물 파일이나 물리적 객체에 대한 정보는 콘텐트 정보이다. 출처 정보, 맥락 정보, 참조 정보, 고정 정보, 접근권한 정보 등은 장기보존 메타데이터에 포함되며, 그 외 진본확인 정보, 이력정보 등을 추가적으로 포함할 수 있다. 장기보존패키지를 논리적으로 구성하기 위해서는 패키징 정보에 DOI 시스템 등과 같이 객체 위치 독립적인 식별체계를 기반으로 할당된 식별자를 반드시 포함하고, 관리하여야 한다.

5.2 패키징 정보

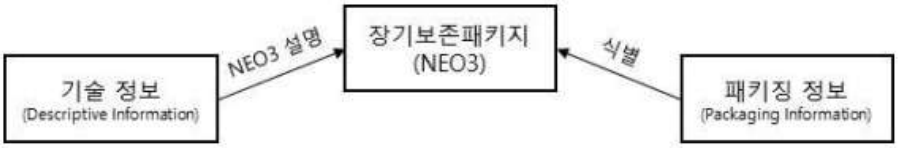


그림 7 - NEO3 기술 정보 및 패키징 정보

패키징 정보는 **그림 7**처럼 장기보존패키지를 구성하는 콘텐트 정보와 장기보존 메타데이터를 연결하는 메타데이터와 구성정보를 포함하는 정보로 NEO3 장기보존패키지의 활용과 계층적으로 구성된 요소들의 목록화를 제공한다.

패키징 정보는 기본적으로 장기보존패키지 내에 포함되는 모든 정보 구성요소들을 하나로 함께 묶는 역할을 한다. 원문 및 보존포맷에 대한 콘텐트정보와 장기보존 메타데이터를 논리적으로 묶어주고 장기보존패키지에 포함되는 추가 부속 파일들에 대한 정보를 함께 묶어 목록화하는 역할을 한다. 또한 장기보존패키지가 수정될 경우 이력정보에 대한 경로 또는 식별자가 포함되어 장기보존패키지 내부에 포함된 파일들에 대한 무결성 확인 및 진본확인을 위한 정보로 활용된다.

기술정보는 장기보존패키지에 대한 메타데이터로 전자기록물 장기보존 기능을 수행하는 기록물관리시스템 등에 검색이나 추출 기능을 적용하는 경우 활용될 수 있도록 선택적으로 추가할 수 있다.

비고 NEO3 장기보존패키지의 기술정보는 선택적으로 포함될 수 있으며, 장기보존패키지는 논리적(logically) 아니면 실제적(actually)으로 생성될 수 있고 물리적 객체로 존재하는 비전자기록물의 경우도 포함될 수 있다.

5.3 콘텐트 정보

5.3.1 구조

콘텐트 정보는 장기보존패키지의 컴포넌트로서 **그림 8**처럼 NEO3 장기보존패키지에 포함되는 원문과 보존포맷에 대한 정보를 포함한다. 두 구성요소를 콘텐트 정보 내에서 함께 보존하며, 보존포맷은 원문의 구조적 정보, 의미적 정보 등 생산 당시 전자기록물의 특성 정보를 보존하여 향후 하드웨어나 소프트웨어의 노후화나 매체의 퇴화 및 기술 변화와 상관없이 원문과 동일한 수준으로 재현하기 위해 포함된다.

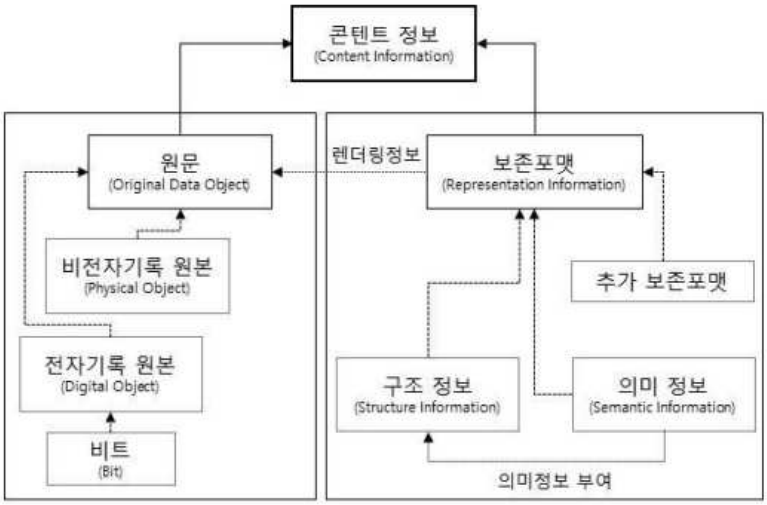


그림 8 - NEO3 콘텐트 정보

5.3.2 원문

원문은 생산자가 생산 또는 접수한 기록물 원본(진본)을 뜻하며 업무활동의 증거로서 법적가치를 가진다. 전자기록물의 경우 특성상 다수의 생산자에 의해 시간 및 공간에 제한 없이 여러 개의 복본을 생산 가능하다.
그러므로 전자기록물은 위변조가 쉽게 이루어질 수 있으며 계속 변화한다. 따라서 진본확인을 위해, 적법한 절차에 따라 관리되어 온 것인지에 대한 과정 등이 메타데이터를 통해 관리되어야 한다.
또한, 기록물 생산 당시의 모습이 변하지 않고 그대로 존재한다고 확인할 수 있어야 진본성을 가지고 있다고 할 수 있다. 그러므로 원문을 장기보존패키지에 포함하여 진본성을 유지한다.

5.3.3 보존포맷

보존포맷은 원문이 생성된 당시의 애플리케이션이 없어도 해당 원문의 내용과 외형을 그대로 재현하여 내용보기를 가능하게 하거나 원문의 내용과 맥락정보 등 유형별 중요 정보를 추가적인 타 기술에 의존하지 않거나 표준화되고 공개된 방식으로 기술적 복원이 가능하게 하는 포맷이다. 즉, 보존포맷은 기록이 생산되었을 때의 기록물(원문)의 의미, 구조 등 기록의 특성을 유지하며, 하드웨어나 소프트웨어의 노후화나 매체의 퇴화 및 기술발전으로 인한 데이터 포맷의 변화에 상관없이 전자기록물을 보존하여 원본이 손상되어

접근이 불가하거나 어려운 경우, 이용자가 원문과 유사한 전자기록물을 접근하여 활용하기 위한 변환된 포맷이다.

원문을 변환할 적합한 보존포맷이 지정되지 않았거나 에뮬레이션 등 다른 보존방식에 의해 재현이 가능한 경우 장기보존패키지에 보존포맷을 포함하지 않을 수 있다. 원문이 장기보존에 취약한 포맷이며 적합한 보존방식이 없는 경우 영구기록물관리기관 등은 가급적 빠른 시일 내에 보존대안을 마련할 수 있도록 노력하여야 한다.

기록물관리기관이 장기보존패키지에 이미 포함되어 관리되고 있는 보존포맷을 다른 보존포맷으로 대체하거나 병행 관리하고자 하는 경우 행위자, 일시, 사유 등을 장기보존 메타데이터에 추가하여 관리할 수 있어야 한다. 이와 관련한 세부 사항 등은 관할 영구기록물관리기관이 정한 방식을 따른다.

5.4 메타데이터

5.4.1 구조

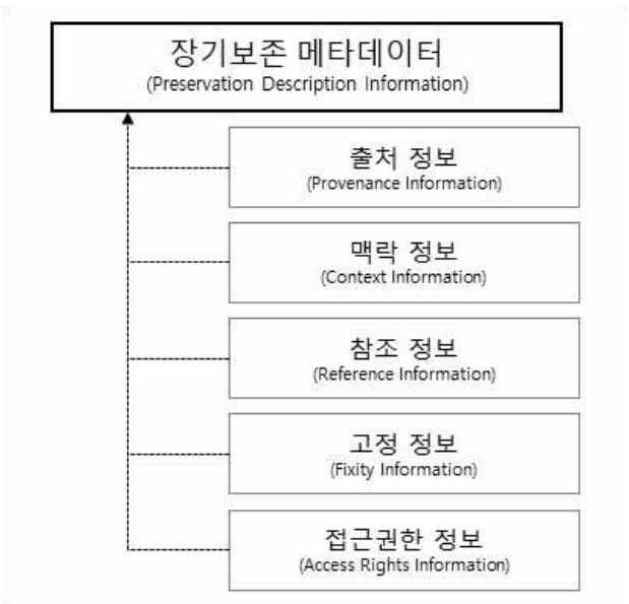


그림 9 - NEO3 장기보존 메타데이터

전자기록물의 장기보존을 위해서는 기록물 원문과 함께 기록물의 생산부터 관리, 보존에 이르는 전 과정을 기술(記述)한 정보를 보존해야 한다. OAIS

정보 패키지는 이를 보존기술정보라고 하며, 이 표준에서는 이러한 장기보존 메타데이터로 지칭한다. 장기보존 메타데이터는 **그림 9**와 같이 출처 정보, 맥락 정보, 참조 정보, 고정 정보, 접근권한 정보 등을 포함한다.

장기보존패키지에서 관리되어야 하는 정보는 다음과 같이 구분할 수 있다.

첫째, 장기보존패키지 자체에 대한 정보인 장기보존패키지명, 버전정보 등이다. NEO3는 XML문서의 계층 구조상으로 NEO2의 루트 요소(Root Element)보다 상위요소로 NEO 루트 요소가 있어 패키징 정보, 장기보존 메타데이터, 콘텐트 정보가 동일한 계층에 존재한다.

둘째, 기록물관리의 전 과정에 관한 메타데이터로서, 기록물철 메타데이터, 기록물건 메타데이터, 컴포넌트와 같은 메타데이터가 계층별로 포함된다. 장기보존패키지에 관리되어야 하는 기록관리 메타데이터는 'NAK 8 기록관리 메타데이터 표준'을 참조하며, 기관 환경에 따라 기록관리 메타데이터 표준에서 제시되지 않은 메타데이터의 추가 및 적용에 관한 사항은 관할 기록물관리기관과 협의하여 적용한다. 단, 이 표준은 다양한 기록물 유형에 따라 관리되어야 하는 메타데이터의 경우 확장 가능한 상위 요소 'ExtraMetadata'를 기록물철, 기록물건 메타데이터 하위에 제공하므로 확장된 XML문서(XML Fragment Document) 및 확장된 요소에 대한 XML 스키마를 포함하여 확장 및 검증 가능한 유연한 구조를 제공한다. 다만, 기록물관리기관이 공통적으로 사용하여야 하는 메타데이터 요소의 추가·변경이 필요한 경우 중앙기록물관리기관의 장과 협의하여야 한다.

셋째, 장기보존패키지에 포함되는 전자기록물 원문 및 보존포맷, 장기보존 메타데이터에 대해 진본확인을 위한 정보(전자서명 정보 등)가 존재할 수 있으며 존재하는 경우, 해당 정보는 패키징 정보에 연결 및 검증 정보를 포함해야 한다.

넷째, 장기보존패키지가 재변환되어 이전 버전의 장기보존패키지를 병행 관리하여야 하는 경우, 재변환 전과 후의 관계에 관한 정보가 추가될 수 있다. 이 경우, 해당 정보는 패키징 정보에 연결 및 검증 정보를 포함해야 한다.

다섯째, 이 표준은 OAIS 참조모델에서 정의한 구성요소인 보존기술정보

(Preservation Description Information), 콘텐트 정보(Content Information), 기술 정보(Descriptive Information)를 별도의 파일로 분리하지 않고 Metadata.xml 내에 통합하여 보존한다. 장기보존 메타데이터는 기록의 출처, 맥락, 참조, 고정, 접근권한 등 장기보존을 위한 기술 정보를 제공하고, 콘텐트 정보는 보존 대상 전자기록물의 내용과 구조·의미 정보를 포함하며, 기술 정보는 기록물의 검색 및 접근을 위한 정보이다.

장기보존패키지의 변경 전과 후에 대한 이력 파일에 대해서는 패키징 정보에 식별자와 해시정보 등이 포함된다. 각 수정 이력 항목별 보존된 패키징 정보는 변경 전 버전의 기록물의 무결성을 검증할 때 활용되며 동시에 변경 후 이력 버전에 대해서도 함께 포함된 해시 값을 통해 연쇄적으로 검증이 이루어져 체인 형태로 연결된 검증이 이루어지게 된다.
앞서 설명한 장기보존 메타데이터는 하나의 파일 또는 여러 개의 파일로 관리될 수도 있다. 이는 장기보존패키지를 생성하는 방식에 따라 다르게 구성된다. NEO3의 경우, 논리적으로 구분된 패키징 정보, 장기보존 메타데이터, 콘텐트 정보 등을 하나의 XML문서 내에 포함하여 장기보존패키지 데이터를 간단하게 처리할 수 있도록 구성한다. 자세한 내용은 **부속서 A(참고) NEO3 장기보존패키지 구성 사례**를 참고한다.

5.4.2 장기보존 메타데이터 구성

기존 NEO2의 경우 기록관리 메타데이터와 장기보존 메타데이터를 3계층, 2계층으로 서로 다른 계층으로 연결되어 매핑되었다. 적용되는 기록계층은 각각 **그림 10**과 같이 매핑된다.

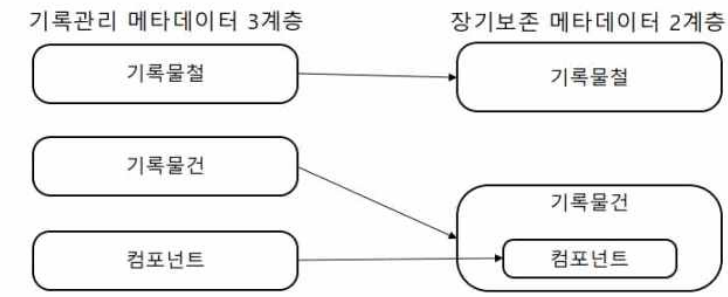


그림 10 - NEO2 메타데이터 매핑

그러나 NEO3의 경우 기록관리 메타데이터의 계층과 장기보존 메타데이터 계층은 차이가 발생하지 않도록 구성된다. 기록물철, 기록물건, 컴포넌트는 **그림 11**과 같이 각각 모두 동일한 계층으로 장기보존패키지에 포함되며, 구성정보는 패키징 정보 또는 장기보존 메타데이터에 디렉토리로 구조화되어 보관된다.

그림 11 - NEO3 메타데이터 매핑

또한, 기관의 환경, 기록물의 특성 등에 따라 'NAK 8 기록관리 메타데이터 표준'에서 정의하지 않은 기록물 철과 건 장기보존 메타데이터를 추가로 관리할 필요가 있는 경우, **그림 12**처럼 메타데이터 하위에 확장 가능한 'ExtraMetadata'라는 이름의 요소에 유연하게 확장하여 추가할 수 있다. 확장, 추가된 메타데이터는 추가된 구조에 대한 XML스키마를 포함하여 유효성 검증을 가능하도록 할 수 있다.

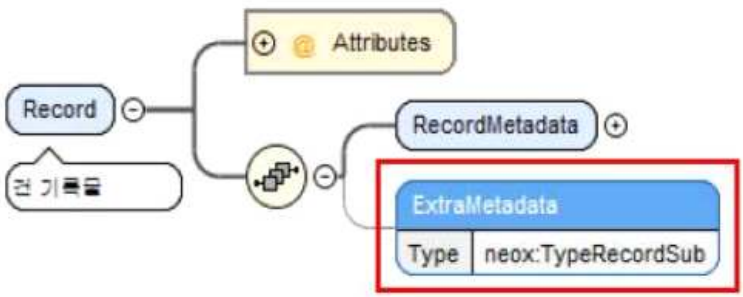


그림 12 - 기록물건 자기보존 메타데이터
기록유형별 확장을 위한 'ExtraMetadata' 요소 포함
(기록물철도 동일)

5.4.3 기록물철 장기보존 메타데이터 구성

기록물철 장기보존 메타데이터는 **그림 13**과 같이 장기보존패키지 객체에 대한 전체적인 정보를 기술하는 객체메타데이터(M5), 기록물의 정보를 표현하는 객체콘텐트(M9)로 구성된다.

· 객체메타데이터(M5)는 객체유형, 객체유형 설명, 객체 생성일시의 요소를 가지며 객체유형의 값은 기록물철이다.

· 객체콘텐트(M9)는 기록물철 메타데이터(M217)로 구성되어 있다.

· 기록물철 메타데이터(M217)는 생산자, 기록식별자, 기록물명, 기술, 주제, 전자기록물여부, 저장정보, 분류, 생산이력, 보존기간정보, 보존장소, 권한, 관리이력, 보존이력, 관계정보의 15여개의 상위요소를 가진다.

이 표준에서 채택한 진본확인 기술인 전자서명 정보 생성은 기록물철 장기보존패키지의 장기보존 메타데이터를 대상으로 한다. 또한 장기보존패키지에 존재하는 모든 컴포넌트들의 해시 값, 식별자 등을 포함하는 콘텐트정보가 존재하는 경우 해당 정보에 대한 전자서명을 통해 진본성 등을 확인할 수 있다.

그림 13 - 기록물철 장기보존 메타데이터 구성

비고 1 NEO3 장기보존패키지에 포함되는 진본확인 정보, 원문 및 보존포맷은 디렉토리 내에 계층적으로 관리되고 ZIP64 등으로 압축 또는 압축 없이 하나의 디지털 객체로 묶여서 보관되는 기술을 활용할 수 있어 별도의 인코딩이나 처리가 불필요하며, 디렉토리 내의 각종 부속 파일의 논리적 연계 정보가 장기보존 메타데이터에 포함되어 실제적으로 아니면 논리적으로 캡슐화가 가능하다.

비고 2 이 표준에서는 진본확인 기술 중 전자서명을 채택하여 적용하는 것을 기본으로 하며, 전자서명에 대한 자세한 사항은 '5.5 진본확인 기술'을 참조한다. 영구기록물관리기관 등은 기관 환경, 기록물 특성 등에 따라 전자서명 외의 진본확인 기술을 적용할 수 있다.

비고 3 객체콘텐트(M9)는 'NAK 31-1:2022(v2.3) 전자기록물 장기보존패키지 기술규격-제1부: XML로 포맷화된 방식(NEO2)'의 객체컨텐츠(M9)와 동일한 요소로 이 표준에서의 용어의 일관성을 위해 명칭을 변경한다. 5.4.3 기록물건 장기보존 메타데이터 구성에도 동일하게 적용된다.

5.4.4 기록물건 장기보존 메타데이터 구성

기록물건 장기보존 메타데이터는 **그림 14**와 같이 장기보존패키지 객체에 대한 전체적인 정보를 기술하는 객체메타데이터(M5), 기록물의 정보를 표현하는 객체콘텐트(M9) 메타데이터로 구성된다.

- 객체메타데이터(M5)는 객체유형, 객체유형 설명, 객체 생성일시의 요소를 가지며 객체유형의 값은 기록물건이다.
- 객체콘텐트(M9)는 기록물건메타데이터(M11)과 컴포넌트(M164)로 구성되어 있으며, 컴포넌트(M164)는 컴포넌트메타데이터(M165)으로 구성되어 있다.

그림 14 - 기록물건(컴포넌트 포함) 장기보존 메타데이터 구성

기록물건 메타데이터(M11)는 생산자, 기록식별자, 기록물명, 기술, 주제, 전자기록물여부, 저장정보, 분류, 생산이력, 보존기간정보, 권한, 공개, 관리이력, 보존이력, 관계정보 등 16개의 메타데이터 상위요소를 가진다. 기관의 환경, 기록물의 특성 등에 따라 기록물철과 마찬가지로 기록물건 메타데이터를 추가로 관리할 필요가 있는 경우, 장기보존 메타데이터 하위에 확장 가능한 'ExtraMetadata'라는 이름의 요소에 유연하게 확장하여 추가할 수 있다. 확장 추가된 메타데이터는 관련 XML스키마에 추가하여 유효성 검증을 가능하도록 할 수 있다.

기록관리 메타데이터의 상위요소 중 유형, 포맷, 저장매체, 크기, 위치를 '저장정보'의 하위요소로, 일시는 '생산이력'의 하위요소로 분산하여 매핑하는 NEO2와 달리 각 컴포넌트 별 연관 메타데이터를 하나의 상위요소 하에 포함하여 정보의 가독성을 높이고, 전자기록물 장기보존 기능을 수행하는 기록물관리시스템에서 효율적으로 처리할 수 있도록 한다.

컴포넌트 메타데이터(M165)는 컴포넌트 제목, 컴포넌트 유형, 컴포넌트 보존이력, 컴포넌트 관계정보, 장기보존패키지 내의 상대경로, 컴포넌트 파일의 해시 값, 파일의 크기 등 메타데이터 요소를 가진다.

컴포넌트 객체는 실제 파일의 장기보존패키지 내의 상대경로 또는 유일한 ID 및 해시 값, 크기 등 기록물에 대한 접근 및 검증 가능한 메타데이터를 포함해야 한다.

컴포넌트 객체에 대응되는 실제 원본 또는 부속 파일은 생산시스템의 정해진 규칙에 따라 계층적 폴더 구조로 별도의 인코딩 없이 독립된 파일들로 장기보존패키지 디렉토리 구조 내에 포함된다.

기록물건의 전자서명 메타데이터는 기록물철의 전자서명 정보와 동일하다.

5.5 진본 확인 기술

5.5.1 일반사항

장기보존패키지는 원본 기록물에 대하여 특정 시점을 기준으로 진본성을 검증하고 해당 시점의 원본 기록물이 손상되지 않았다는 무결성 검증을 위해 진본확인 기술을 적용하여야 한다. 이 표준에서는 기록물의 진본성, 무결성 유지를 위해 전자서명, 해시기법 등의 기술을 적용하며, 전자서명 정보를 장기보존패키지에 포함할 수 있다. 영구기록물관리기관 등은 진본확인을 위한 기술을 달리 정할 수 있다.

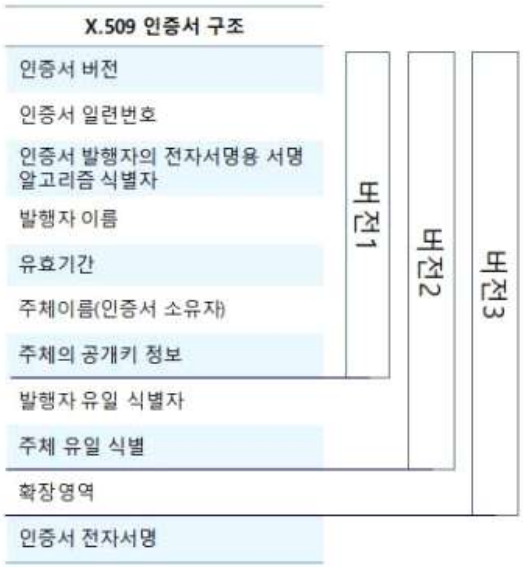


그림 15 - X.509 인증서 구조

NEO3 전자서명은 ITU-T의 공개키 기반(PKI)으로 적용하며, **그림 15**과 같은 PKI 기반 X.509 전자서명용 인증서가 필요하다. 이렇게 생성된 전자서명 정보 및 인증서는 장기보존패키지에 함께 포함되어 해당 기록물의 서명자(서명주체)를 확인할 수 있고 전자서명 시점에 대상 기록물에 대한 진본성과 기록물이 변경되지 않은 무결성을 확보할 수 있다.

디렉토리로 구조화된 NEO3 장기보존패키지 내의 전자서명 대상은 장기보존 메타데이터 파일이며, 원문 및 보존포맷 등 콘텐트 정보 파일에 대한 파일별 해시정보는 장기보존 메타데이터 또는 패키징 정보에 이미 포함되어 있어 전체 장기보존패키지에 대한 검증이 연쇄적으로 가능하게 된다.

비고 전자서명 XML 정보는 장기보존패키지에 포함된 인증서(GPKI, PKI 등)를 이용해 원본 장기보존 메타데이터의 해시 값을 기반으로 생성하며 전자서명 확인 시점에 인증서에 포함되거나 포함된 인증서 공개키로 해당 전자서명 XML 파일에서 원본 해시 데이터를 복호화하여 얻고, 전자서명 검증 시점의 장기보존 메타데이터에 대한 해시 값을 계산하여 두 해시 값을 비교하여 검증할 수 있다.

5.5.2 인증정보

이 표준에서는 장기보존패키지의 진본확인 기술 중 하나인 전자서명 방식을 적용하였다. 이 방법은 장기보존패키지에 포함된 전자서명 XML과 인증서를 이용하여 해당 기록물이 변경되지 않았음을 검증함으로써 전자기록물의 진본성과 무결성을 유지해 주는 방법이다.

그림 16 - GPKI 인증서 기반 인증정보 생성 및 활용 방법

그림 16은 GPKI 인증서 기반 인증정보 생성 및 활용 방법을 보여주고 있다. 전자서명 생성은 'KS X 6033 전자서명 표준'에 따라 인증정보를 생성하거나 행정전자서명 인증관리 센터에서 배포되는 공공 GPKI API가 지원하는 전자서명 함수를 통하여 진행될 수 있다. 전자서명 대상은 앞서 기술한 바와 같이 장기보존 메타데이터를 대상으로 한다. 또한 인증정보 생성을 위한 인증서는 용도가 전자서명용으로 생성된 인증서를 활용하도록 한다. 전자서명용으로 발급된 GPKI 기관용 인증서(기록물 생산기관 또는 기록물 관리기관 인증서 등)를 사용할 수 있다.

5.5.3 전자서명 인증서의 장기검증

X.509 인증서를 사용하여 전자서명 인증정보를 생성하는 경우, 해당 인증서가 도난 등 사고로부터 유효한지 또는 인증서 기간이 만료가 되었는지 확인이 필요하다.

PKI기반 인증서 발급 기관(CA:Certificate Authority)에서 폐기되거나 유효하지 않은 인증서에 대해서 인증서 폐기 목록(CRL:Certificate Revocation List)을 주기적으로 발급하게 된다. GPKI 인증서의 경우 인증서 기간이 만료되더라도 관련 CA기관에서 발행한 CRL 기반으로 인증서가 위변조되지 않았는지 검증이 가능하다.

전자기록물을 장기보존하는 경우 기록물의 진본성 및 무결성을 주기적 또는 필요한 시점에 검증할 수 있어야 한다.

이 표준에서 진본확인을 위한 기술로 적용한 전자서명에 대해 지속적으로 검증할 수 있어야 하나 전자서명 검증을 위해 필요한 인증서의 유효 가능 기간은 기록물을 보존해야 하는 기간에 비해 짧다. 행정전자서명의 경우 행정전자서명인증관리센터에서 발급하는 인증서 유효기간 범위 내이다.

유효기간이 지난 인증서를 기반으로 한 전자서명은 효력을 상실하게 됨으로써 기록물의 위변조 여부를 확인하기 어렵다. 따라서 인증서의 유효기간이 지나더라도 기록물의 전자서명에 사용된 인증서의 유효성을 계속적으로 검증하기 위해서는 별도의 체계가 필요하다.

비고 전자서명 인증서를 장기 검증할 수 있는 방법은 “NAK 32-1:2022(v1.2) 전자기록물 전자서명 인증서 장기검증 기술규격”을 참조하거나 전자서명 시 사용한 인증서를 발급한 인증 기관(CA)에서 전자서명하여 발급하는 인증서 폐기 목록(CRL)에 의해서 비교하여 확인할 수 있다. 인증서가 폐기되는 이유는 RFC3280/RFC5280(Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List Profile)으로 확인할 수 있다.

5.6 장기보존패키지 변경이력 관리

장기보존패키지가 수정되는 경우는 관리이력이나 바이러스 검사, 오류 검증으로 인한 메타데이터 수정, 전자서명 생성, NEO2에서 NEO3로 장기보존패키지 생성방식 변환, 새로운 보존포맷 추가 등으로 NEO3 장기보존패키지 내에서 실제 수정되는 파일은 장기보존 메타데이터 파일, 진본확인 정보(전자서명 파일, 전자서명 인증서 등) 파일이다.

표 1 - NEO3 변경 전후 디렉토리 구조 비교

변경 전	변경 후
장기보존패키지 최상위폴더 Change History 파일 폴더 Original Content 파일 폴더 Preservation Format 파일 폴더 Certification.cer 보안 인증서 Metadata.xml XML Document Metadata.xsd XML Schema mimetype 파일 PackagingInformation.xml XML Document PackagingInformation.xsd XML Schema Signature.xml XML Document Change History 디렉토리 하위에 수정 전 추가 .. 파일 폴더 Certification_v0.cer 보안 인증서 Metadata_v0.xml XML 파일 Signature_v0.xml XML 파일	장기보존패키지 최상위폴더 Change History 파일 폴더 Original Content 파일 폴더 Preservation Format 파일 폴더 Certification.cer 보안 인증서 Metadata.xml XML Document Metadata.xsd XML Schema mimetype 파일 PackagingInformation.xml XML Document PackagingInformation.xsd XML Schema Signature.xml XML Document Change History 디렉토리 하위에 수정 전 추가 .. 파일 폴더 Certification_v0.cer 보안 인증서 Certification_v1.cer 보안 인증서 Metadata_v0.xml XML 파일 Metadata_v1.xml XML 파일 Signature_v0.xml XML 파일 Signature_v1.xml XML 파일

장기보존패키지의 장기보존 메타데이터가 수정되는 경우, 표 1과 같이 최신 정보는 현재의 장기보존 메타데이터 파일에 반영되고, 수정 전 정보는 별도의 버전을 부여하여 각각 별도의 파일로 함께 보존한다.

또한 장기보존패키지 내의 디렉토리 계층 및 위치 등에 대한 변경사항은 패키징 정보가 존재하는 경우 관련 변경 정보 수정이 병행되어야 한다. 이렇게 버전 별로 별도 파일로 변경이력을 관리함으로써 변경 내용에 대해 수정된 메타데이터 버전 간의 비교, 각 버전별 진본확인 기술 적용 등을 쉽고 효율적으로 처리할 수 있게 한다. 기록물에 대한 메타데이터 정보 수정 시 패키징 정보에 추가되는 변경 일시, 버전 등의 구분정보 등을 기준으로 명명 규칙을 수립하고 이를 기준으로 수정 파일을 생성하여 보존하도록 한다.

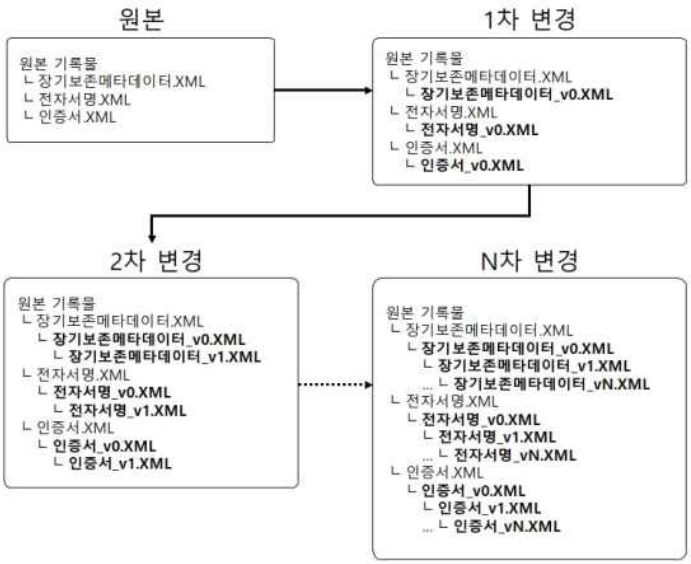


그림 17 - 장기보존패키지 변경 이력 관리 방법

그림 17는 장기보존패키지 내의 파일이 변경되는 경우 N차까지 버전을 부여하여 수정 전과 후에 대한 이력을 개별 파일로 추가 관리 방법 및 명명 규칙 사례를 보여준다. 장기보존패키지 내 변경 이력 파일들의 전후 관계는 파일명 등으로 유추해서는 안된다. 변경 이력 정보의 생성 시점은 패키징 정보에 포함된 생성 시점 정보를 통해서 확인해야 한다.

이 표준에서는 장기보존패키지 변경이력 관리의 효율성을 높이기 위해 데이터 무결성과 중복 최소화 원칙을 기반으로 한 개선 방안을 제시한다. 이 방안은 변경 발생 시의 기록 구조, 이력 폴더 관리 방식, 복원 절차를 포함하며 장기보존패키지의 최신성과 신뢰성 확보를 위한 변경이력 관리 방안이다. 변경이력 관리에 대한 구체적인 방식은 제7장에서 상세히 기술한다.

5.7 도큐멘테이션

Documentation은 장기보존 메타데이터 외에 기록물의 내용과 구조를 이해하거나 활용하기 위한 문서, 자료, 데이터 등의 부가적인 정보가 존재하는 경우 필수적으로 요구한다. 또한 Documentation은 전자기록물의 유형과 관계없이 이러한 부가 정보를 포함할 수 있도록 구성하여야 한다.

6 기록물 유형에 따른 장기보존패키지 생성 과정

6.1 장기보존패키지 생성 과정

NEO3 장기보존패키지를 하나의 물리적인 파일로 생성하기 위해서 ZIP64등의 기술을 활용할 수 있다. 기술정보, 콘텐트 정보, 장기보존 메타데이터, 패키징 정보에 대한 기술정보, 진본확인 정보 등 **그림 6**에서 설명된 장기보존패키지를 구성하는 다양한 개념적 요소들을 여러 개 또는 하나의 XML 파일로 분리, 통합하여 구성할 수 있다.

그림 18는 장기보존패키지의 개념적 요소 항목들을 계층적 관계에 따라 나누어 설계한 XML 스키마 기반의 XML 문서 파일로 묶어서 구현된 NEO3 장기보존패키지 구조이다. 패키징 정보는 PackagingInformation.xml, 그리고 장기보존 메타데이터, 콘텐트 정보, 기술 정보는 Metadata.xml의 이름을 가진 XML 형식의 파일로 생성된다. 패키징 정보, 장기보존 메타데이터, 콘텐트 정보는 모두 필수적인 반면, 기술 정보는 선택적으로 포함시킬 수 있다. 원문 및 보존포맷들은 각각 패키징 정보에서 정해진 파일명으로 설정된 각각의 위치에 저장된다. Metadata.xml 내 콘텐트 정보는 원문과 보존포맷과 연관된 부가정보(예: 파일명, 크기, 버전, 형식 등)를 포함할 수 있다.

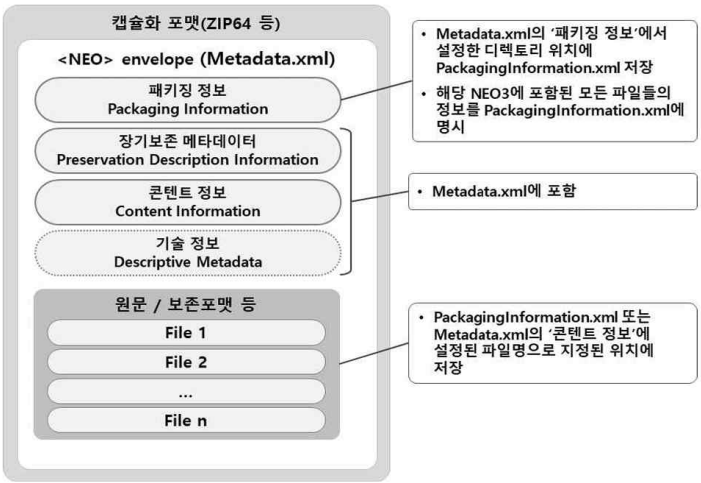


그림 18 - NEO3 캡슐화 구조

장기보존패키지(NEO3)의 전체 구조는 **그림 19**와 같이 최상위 요소(root

element)인 NEO3와 4개의 하위 요소로 구성되어 있고 XML 스키마로 설계되었다. 그러므로, 장기보존패키지 개념적 구성을 상호 연관관계와 계층적 구조에 따라 하나의 XML 구조로 통합 설계하거나 필요에 따라 개별 항목으로 분리하여 구성할 수 있다.

비고 ISO 14721 OAIS 참조모형의 정보 패키지를 참고하여 구현한 사례는 E-ARK 등이 있다. E-ARK(European Archival Records and Knowledge Preservation)는 유럽위원회(European Commission, EC)의 지원을 기반으로 2014년 2월부터 전자기록물 장기보존을 위해 추진된 프로젝트이다. 영국, 독일, 포르투갈, 덴마크, 노르웨이, 헝가리, 에스토니아, 스페인, 포르투갈 등의 아카이브 기관 및 대학 등이 참여하고 있으며, 데이터세트 유형의 전자기록물 장기보존 도구 개발 및 SIP, DIP, AIP 등의 정보패키지 표준화를 수행하고 있다.

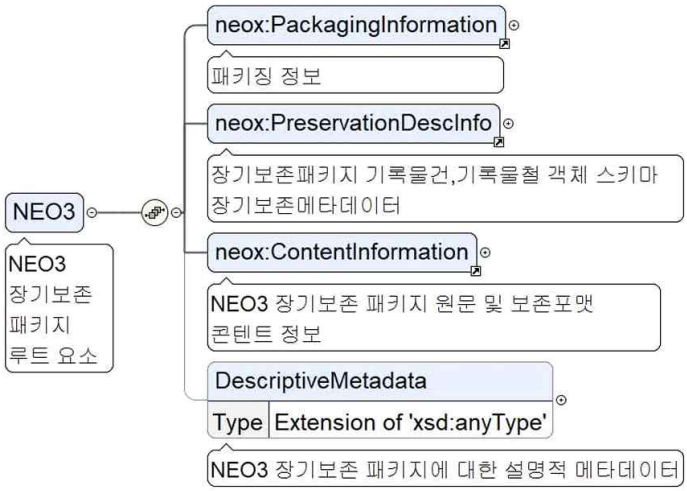


그림 19 - NEO3 패키징 정보, 장기보존 메타데이터, 콘텐트 정보 통합 구성 XML 스키마 구조

그림 20은 NEO3 통합 구성 스키마와 장기보존패키지 개념적 요소의 관계정보이다. NEO3 개념적 구조의 최상위 수준의 개념인 장기보존패키지(NEO3)는 NEO3 XML 스키마에서 NEO3 요소로 구현된다. 그리고 장기보존패키지(NEO3)를 식별하는 패키징 정보는 neox:PackagingInformation 요소로, 장기보존패키지(NEO3)를 설명하는 기술 정보는 DescriptiveMetadata 요소로, 콘텐트 정보는 neox:ContentInformation 요소로, 장기보존 메타데이터는

PreservationDescInfo 요소로 각각 대응되어 설계되었다.

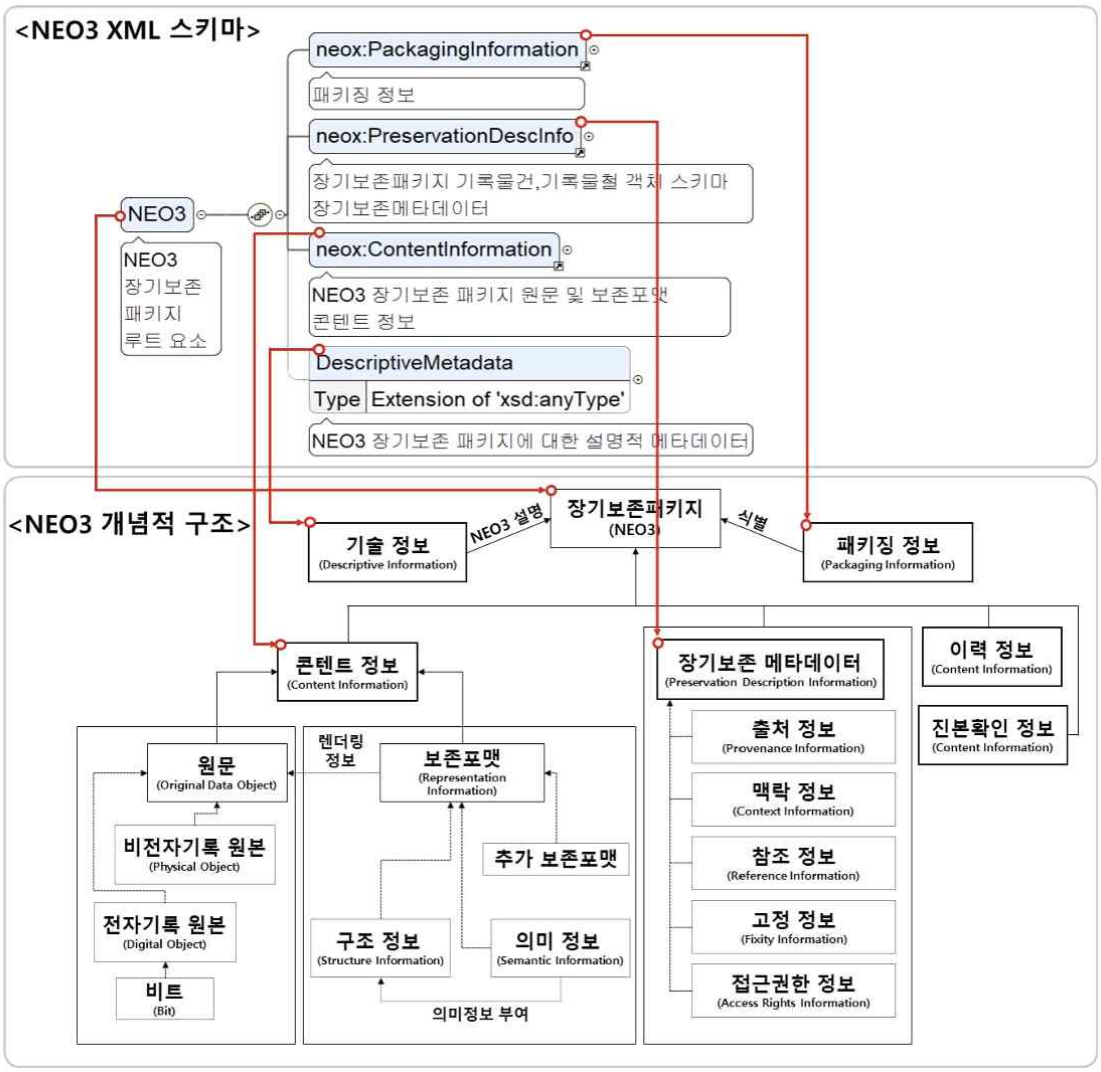


그림 20 - NEO3 개념적 구조 기반 XML 스키마 설계

장기보존패키지는 **그림 21**의 단계에 따라 생성되어야 한다. 첫째, 패키징 대상의 범위를 설정하고 수집 정책을 수립하여 패키징 대상 기록물과 메타데이터를 정의하여야 한다. 둘째, 전자기록물 정합화 단계에서는 원문과 보존포맷 파일을 구성하고 상호 연계해야 한다. 셋째, Metadata.xml 파일에 원문 및 보존포맷 파일의 목록과 해시값을 수록하여 참조일관성을 확보하는 기록관리 메타데이터 확정 단계를 수행해야 한다. 넷째, Metadata.xml을 입력값으로 하여 서명값을 생성하고 Signature.xml을 작성한 후, 인증서 파일(Certificate.cer)을 첨부하는 행정전자서명 단계를 수행하여야 한다. 다섯째,

폴더 구조를 정렬하고 파일 배치를 완료한 뒤 패키지 인벤토리와 해시값을 산출하여 PackagingInformation.xml에 기록하여 장기보존패키지를 최종 확정한다.

그림 21 - 장기보존패키지 생성 과정

각 단계의 구체적인 절차 및 산출물 작성 방법은 기록물 유형별 특성에 따라 구분하여 설명한다.

6.2 문서유형 기록물

생산시스템에서 생산된 문서유형의 전자기록물은 문서 단위로 기록관리 메타데이터가 자동 생성되며, 해당 메타데이터는 생산 시점부터 기록물의 속성 및 관리 이력을 기술한다. 보존기간이 30년 이상인 전자기록물 중 10년 이상 보존된 기록물은 장기보존패키지로 전환하여야 한다. 이때 기존의 기록관리 메타데이터는 장기보존 메타데이터로 변환되며, 부록에 제시된 장기보존 메타데이터 구성에 따라 Metadata.xml 파일의 보존기술정보 요소에 포함한다. 또한 PreservationDescInfo 요소에는 생산 단계에서 생성된 기록관리 메타데이터의 주요 정보를 연계하여 기록물의 생산 및 관리 과정을 반영한다.

문서유형 기록물은 본문과 첨부파일로 구성되며, 이 전체를 기록물의 원문으로 본다. 본문과 첨부파일은 각각 원문 폴더에 포함하며, 생산 당시의 형태와 구조를 유지한 상태로 저장하여야 한다. 첨부파일 또한 원문을 구성하는 요소로 함께 관리한다.

장기보존패키지를 생성할 때 원문이 장기보존에 적합하지 않은 형식으로 생산된 경우에는 표준화된 장기보존용 포맷(예: PDF/A 등)으로 변환하여 보존포맷으로 함께 포함하여야 한다. 생산 당시의 파일은 원문으로 포함하고, 변환된 파일은 보존포맷으로 추가하여 동일 문서의 두 형태를 병행하여 보존한다. 이미 보존포맷으로 생산된 기록물은 원문과 동일한 파일을 보존포맷 폴더에 함께 포함한다. 자세한 내용은 **부속서 A(참고) 장기보존패키지 구성 사례**를 참고한다.

6.3 데이터세트유형 기록물

6.3.1 데이터세트 설정

6.3.1.1 일반사항

장기보존패키지의 구성요소인 원문, 보존포맷, 장기보존 메타데이터의 구조와 내용을 데이터세트 유형에 적합한 방식으로 구체적으로 정의한다. 이를 통해 데이터세트의 특성과 보존 요구에 부합하는 장기보존패키지 구성을 마련한다.

전자기록물 장기보존패키지의 생성 과정에서 데이터세트 및 메타데이터 설정 단계는 패키지의 기본 구조와 관리 기준을 확정하는 선행 절차이다. 이 단계에서는 보존 대상이 되는 데이터세트의 범위를 선정하고, 스냅샷 정책을 수립하여 데이터 추출 및 패키징 시점을 결정하며, PID/SID 식별자와 버전 관리 방식을 설정하여 이력 추적성을 확보하고, 메타데이터 요소를 PackagingInformation.xml과 Metadata.xml에 체계적으로 매핑하는 작업을 포함한다.

6.3.1.2 메타데이터 설정

Metadata.xml은 장기보존패키지에서 반드시 포함되어야 하는 필수 파일이다. **그림 22**와 같이 이 파일에는 장기보존메타데이터(PDI)를 담는 PreservationDescInfo, 콘텐트 정보(CI)를 담는 ContentInformation, 설명 정보(DI)를 담는 DescriptiveMetadata를 포함한다.

Metadata.xml의 PreservationDescInfo 요소는 '행정정보 데이터세트 종합관리 시스템'에서 해당 시스템의 관리기준표로부터 정보를 수집하여 작성하여야 한다. 다만 관리기준표로 모든 정보를 충족할 수 없는 경우에는 자체관리하는 행정정보시스템, 행정정보시스템 산출물, 정보자원관리시스템으로부터 정보를 보완하여 작성할 수 있다.

비고 행정정보 데이터세트 관리기준표에 대한 자세한 사항은 'NAK 35:2020(v1.0) 행정정보 데이터세트 기록관리 기준'을 참조한다.

ContentInformation 요소는 원문과 보존포맷 파일의 목록 및 해시값을 담아야 한다. Metadata.xml에는 이미 기록관리 메타데이터가 포함되어 있기에 해당 요소의 파일 목록과 해시값을 둠으로써 전자서명만으로 원문과 보존포맷의 위·변조 여부를 검증하도록 한다.

DescriptiveMetadata 요소는 PreservationDescInfo 요소의 정보를 추출 또는 요약하여 설정하여야 한다. 이를 통해 아카이브에 보존된 디지털 객체를 고유하게 식별할 수 있도록 지원하고, 이용자가 해당 객체를 효과적으로 검색할 수 있는 조건을 제공한다. 또한 디지털 객체의 맥락, 내용, 형식 등을 이해할 수 있도록 부가 정보를 제공하는 역할을 한다. DescriptiveMetadata 요소는 기관마다 사용하는 시스템의 기술 메타데이터 구조가 서로 다르기 때문에 xsd:anyType으로 설정되어 있다. 이는 기본적인 구조는 표준화하되, 세부 항목은 기관별 환경에 맞게 자유롭게 정의할 수 있도록 설계한 것이다.

작성된 메타데이터는 국제 표준 스키마(XML 기반)에 따라 구조화하며, 시스템을 통해 자동으로 Parameter XML 파일을 생성한다. 생성된 Parameter XML은 표준 스키마에 적합한지 자동·수동 검증 절차를 거쳐야 하고, 관리자는 검증 절차를 거쳐 오류가 없는 경우에만 보존패키지에 포함하여야 한다. 이 검증 절차는 메타데이터의 구조적 일관성과 상호운용성을 보장하기 위해 반드시 수행되어야 한다.

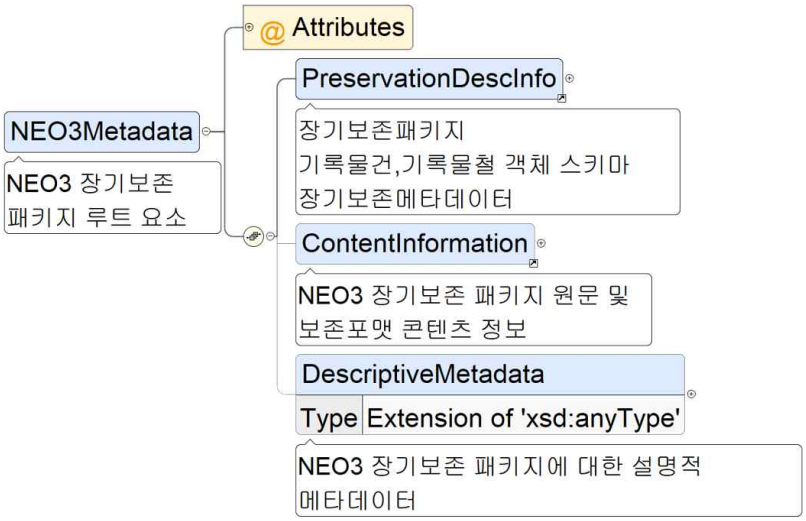


그림 22 - Metadata.xml의 구조

6.3.1.3 데이터세트 설정

데이터세트의 원문은 데이터베이스에서 텍스트 형식의 덤프(dump) 파일로 추출하여야 한다. 또한 원본 데이터를 재현하는 데 필요한 환경 정보가 존재할 경우, 데이터베이스로부터 추출해야 한다. 이러한 절차는 데이터베이스의 구조와 내용을 원형 그대로 보존하기 위한 것이다.

붙임 파일 즉 데이터베이스에서 관리되지 않고 데이터베이스의 특정 필드값과 디스크 스토리지의 파일들이 연결되어 있는 경우 이를 추출해야 한다. 붙임 파일은 기관마다 관리하는 방식이 다르므로 가관별로 알맞은 방식을 채택할 수 있다.

비고 데이터베이스에는 실제 데이터가 저장되지 않고, URL, URN 등의 식별자로만 표시되어 있으며, 별도로 저장되어 있는 파일들을 붙임이라 한다. 업무수행시 첨부자료의 형태로 이러한 붙임들이 대량·대용량으로 저장·관리되고 있는 실정이며, 행정정보 데이터세트의 기록물의 범위에 이 붙임을 구성요소로 정의하고 있으므로, 장기보존패키지에서 수용할 수 있어야 한다.

행정정보시스템의 산출물은 각 기관의 행정정보시스템 관리자 또는 정보자원관리시스템을 통해 수집할 수 있다.

비고 데이터세트의 경우에는 정보시스템 구축 후에 행정정보시스템 산출물들이 생산된다. 데이터세트를 이해하는 데 크게 도움이 되는 정보이므로 장기보존패키지에서 수용할 수 있어야 한다. 대표적인 정보시스템 산출물은 제안요청서, ERD, 시스템 설계서, 사용자 및 운영자 매뉴얼, 화면설계서, 연계시스템 정의서 등이 있다.

6.3.2 데이터세트유형 기록물 추출

장기보존패키지를 생성할 때에는 원문 데이터와 보존포맷 간의 정합성을 확보하여야 한다. 이를 위해 원문(예: DB dump), 보존포맷(데이터세트/xsd+xml, 재현 정보, 붙임 파일) 및 행정정보시스템 산출물 간의 구성 및 연계하여야 한다.

장기보존패키지 구성 시 데이터세트 유형의 전자기록물은 원문, 붙임, 행정정보시스템 산출물, 보존포맷의 네 가지 요소를 중심으로 구성하고, 이들은 정해진 폴더 구조에 따라 체계적으로 배치된다.

먼저, **그림 23**와 같이 원문은 장기보존패키지의 핵심 객체로서 Original Content 폴더에 배치하여야 한다. 원문은 데이터베이스 덤프(DB dump) 또는 기타 전자기록물의 원본 파일 형태로 존재하여야 한다. 붙임은 원문과 연계되는 첨부 파일로, 실제 파일을 보존 대상에 포함시킨 경우, Original Content 폴더에 배치하여야 한다. 또한, 행정정보시스템 산출물은 최상위 디렉토리 수준에 Documentation 폴더를 별도로 생성하여 보관하여야 한다.

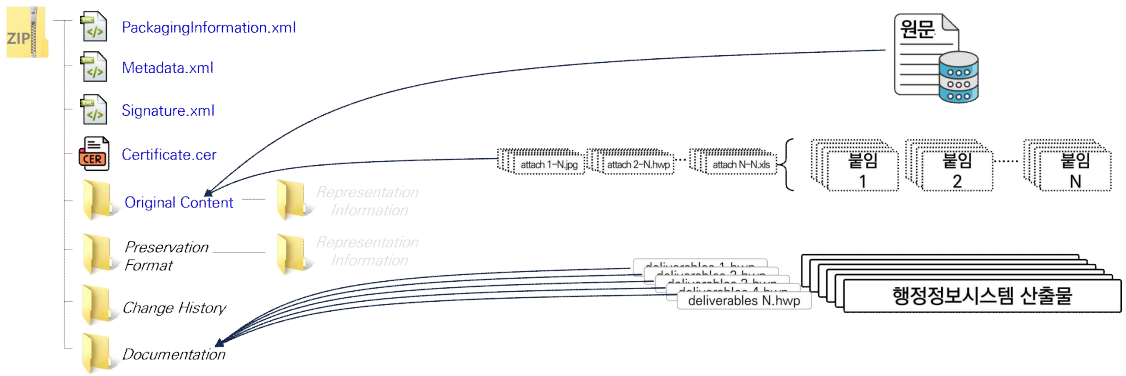


그림 23 - 원문의 위치 및 구성요소

보존포맷은 원문을 변환하여 장기적인 접근성과 상호운용성을 확보하기 위한 파일로 **그림 24**와 같이 Preservation Format 폴더에 배치하여야 한다. 다만 보존포맷이 원본과 동일한 경우에는 별도의 Preservation Format 폴더를 만들지 않는다.

비고 1 전자기록물 보존 포맷 생성시 '전자기록물 보존포맷 선정기준(NAK 37:2022(v1.0))'을 참조한다.

비고 2 데이터세트의 경우, 이러한 보존포맷은 데이터베이스의 구조와 내용을 표준화 및 정규화하여 재현 가능하게 만든 파일로, 일반적으로 dpf(dataset preservation format) 형태로 저장한다. dpf는 덤프 파일이 특정 DBMS 환경에 종속되지 않도록 가공한 보존용 파일로, 스키마 정보(DDL), 데이터 값(DML), 설정·관리되는 사용자 권한 및 접근제어 정보(DCL) 등을 포함한다. 이를 통하여 시간이 지나더라도 운영 시스템 없이 데이터를 복원할 수 있게 된다.

붙임이 보존포맷으로 변환되는 경우에도 동일하게 Preservation Format 폴더에 배치하여야 한다. 다만 보존포맷으로 변환된 붙임 파일이 원본과 동일한 경우에는 별도의 Preservation Format 폴더를 만들지 않는다.

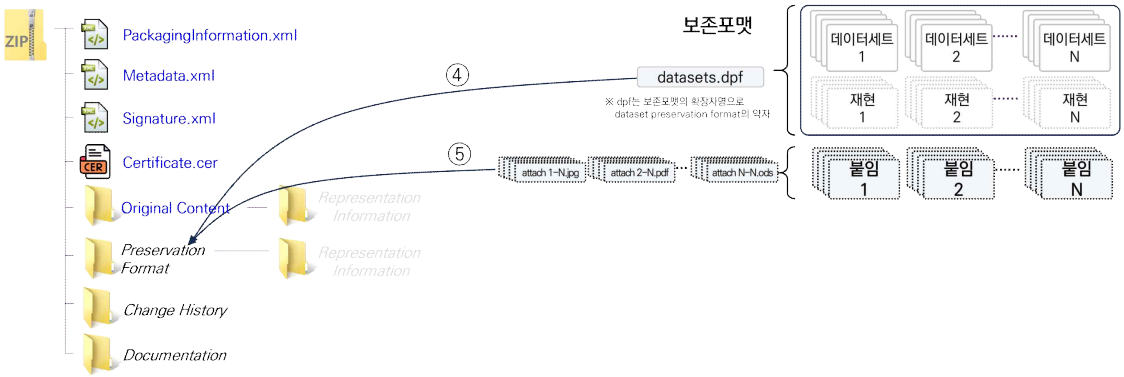


그림 24 - 보존포맷의 위치 및 구성요소

이와 같은 폴더 구조는 NEO3 장기보존패키지의 표준 구성 원칙에 따라 설계하였다. 이를 통해 전자기록물의 원본성과 진본성을 확보하고, 변환된 보존포맷을 통해 장기 접근성을 강화한다.

6.3.3 Metadata.xml 설정

Metadata.xml 파일에 원문과 보존포맷 파일의 목록 및 해시값을 기록함으로써, 패키지 내 모든 파일 간 참조가 정확히 일치하도록 관리하여야 한다. 이를 통해 장기보존패키지의 구성 요소 간 연결 관계를 명확히 하고, 패키지 무결성을 검증할 수 있는 근거를 확보한다.

먼저, PreservationDescInfo 요소를 설정한다. 데이터세트 유형의 경우, PreservationDescInfo는 시스템 단위와 데이터세트 단위로 구분한다. 행정정보시스템 산출물과 관련된 정보는 SystemInfoCon\DeliverableCon XML 요소에 기술하여야 하고, 이 영역에는 설계서, 메뉴구조도, ERD 등을 포함한다. 데이터세트 관련 정보는 DatasetInfoCon\ComponentCon XML 요소에 작성하여야 하고, 여기에는 원문, 보존포맷, 붙임 파일 등의 정보를 모두 포함한다. 이를 통해 패키지 내에서 시스템 산출물과 데이터세트 파일 간의 관계를 명확히 구조화하고, 후속 관리와 재현이 가능하도록 하여야 한다.

다음은 ContentInformation 요소를 설정한다. 이 단계에서는 Original Content 폴더에 포함된 모든 파일을 대상으로 ContentInformation\ContentItem에 값을 부여하여야 한다. 각 파일의 경로, 파일명, 식별자, 해시값 등의 정보를 기록하여 패키지 내 파일이 장기보존 과정에서 변형 없이 유지되고 있음을 검증할 수 있도록 한다.

마지막으로 DescriptiveMetadata 요소를 아래 **그림 25**과 같이 설정하여야 한다.

<NEO3Metadata - DescriptiveMetadata>

```
<DescriptiveMetadata>
  <Title>...</Title>                          <!-- MODS titleInfo/title -->
  <Identifier>...</Identifier>                <!– 기록관리식별자 등 -->
  <Creator>...</Creator>                      <!-- MODS name -->
  <Subject>...</Subject>                      <!-- 반복 가능 -->
  <Description>...</Description>              <!-- 요약 설명 -->
  <DateCreated>...</DateCreated>              <!-- 생산일자 -->
  <Language>...</Language>                    <!-- 언어코드: ISO 639-2 (kor, eng) -->
  <AccessRights>...</AccessRights>            <!-- 공개/제한 여부 -->
</DescriptiveMetadata >
```

그림 25 - DescriptiveMetadata 요소

비고 DescriptiveMetadata 요소는 OAIS 모델의 Descriptive Information 요구사항(식별, 발견, 검색, 접근, 이해)을 충족하고 있다. Dublin Core의 기본 메타데이터 요소를 채택하였고, EAD에서 필요한 항목을 발췌하였으며, METS의 <dmdSec>와 PREMIS의 <object> 요소와 연계 가능하도록 설계되어 있다.

이와 같이 PreservationDescInfo, ContentInformation, DescriptiveMetadata의 세 요소는 반드시 설정하여야 한다. 이를 통해 데이터세트 기록물의 구조와 출처를 명확히 정의하고, 파일의 무결성을 보장하며, 행정정보의 맥락적 이해와 장기적 접근성을 확보할 수 있다. 이러한 절차는 전자기록의 진본성과 신뢰성을 보장하고, 패키지 단위의 표준화된 관리와 재현을 가능하게 한다.

6.4 행정전자서명

6.4.1 Signature.xml

Signature.xml은 장기보존패키지 내에서 장기보존 메타데이터와 장기보존패키지 내부 참조의 변조 여부를 검증한다. Signature.xml에는 Metadata.xml에 대한 전자서명값와 기관인증서(GPKI)를 기록하고, 장기보존패키지 전체 구성요소의 무결성 검증 체인을 유지하는데 사용하여야 한다. Signature.xml은 Metadata.xml의 해시값을 기반으로 생성된 서명정보를 보존함으로써 장기보존패키지의 구성요소가 생성 이후 이관·보존 과정에서 변형되지 않았음을 검증하기 위한 기준으로 기능한다.

Signature.xml은 **그림 26**과 같이 행정전자서명 인증관리센터에서 발급받은 기관인증서(GPKI)를 이용하여 생성하여야 한다. 패키지 생성 시 폴더에 배치한 모든 파일 목록을 PackagingInformation.xml에 기록한 후, Metadata.xml의 전자서명값을 생성하여 Signature.xml에 전자서명값과 기관인증서(GPKI)를 함께 기록하고 폴더에 추가한다. 이후 기록관에서 발행한 전자서명값(①)과 영구기록물관리기관 이관 후 재생성된 전자서명값(②)을 비교함으로써, Metadata.xml의 변조 여부를 확인한다.

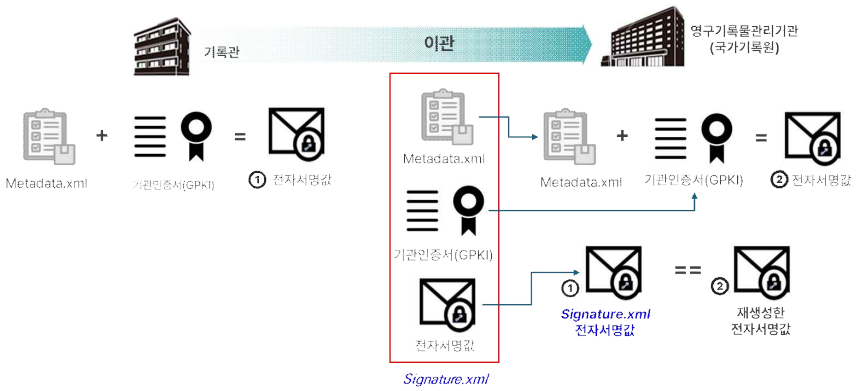


그림 26 - 장기보존패키지 서명 생성 및 진본성 검증 프로세스

6.4.2 Certificate.cer

Certificate.cer은 Signature.xml에 포함된 전자서명값의 진본성을 확인하기 위한 기관 인증서 파일이다. Certificate.cer은 행정전자서명 인증관리센터에서 발급하며, 서명 생성에 사용된 인증기관 정보와 인증서 식별정보를 포함하여야 한다. Certificate.cer은 Signature.xml과 항상 함께 보존되어야 하며, 상호 보완적으로 관리되어야 한다. Certificate.cer은 패키지 내부 메타데이터의 변조 여부를 검증하는 근거로 기능하고, Signature.xml은 해당 서명의 유효성을 입증하는 기준으로 사용한다.

6.5 장기보존패키지 생성

데이터세트 유형 전자기록물의 장기보존패키지를 구성하는 마지막 단계인 패키징 과정은 장기보존패키지 내부의 모든 디지털 객체를 체계적으로 목록화하여야 하고, 이를 기반으로 하나의 패키지 파일로 압축하여 완성하여야 한다.

먼저, **그림 27**과 같이 장기보존패키지에 포함된 모든 디지털 객체에 대하여 생성날짜, 식별자, 해시값, 경로정보를 목록화하여 PackagingInformation.xml를 생성하여야 한다. 이를 통하여 패키지 전체 구성요소를 체계적으로 관리할 수 있도록 하고, 장기보존패키지의 무결성 검증과 재현 과정에서 필수적인 기반 정보로 활용할 수 있다.

비고 PackagingInformation.xml을 통해 패키지의 구성 요소를 명확히 식별하고 검증 가능하게 함으로써, 장기보존 중에도 정보의 진본성과 무결성을 유지할 수 있는 기반을 제공한다. 이를 통해 기록물의 이관, 보존, 재현 및 활용 과정 전반에서 신뢰성과 일관성을 확보할 수 있다.

전체 파일을 포함한 패키지를 압축하여 하나의 ZIP 파일로 생성하여야 한다. 이렇게 생성된 ZIP 파일은 장기보존패키지의 최종 결과물로, 이관 및 보존, 활용의 단위로 사용한다.

이와 같은 절차는 NEO3 장기보존패키지 표준에서 규정하는 핵심 구성 방식으로, 디지털 객체를 개별 파일 단위가 아닌 패키지 단위로 관리하고 장기적으로 보존하기 위한 목적을 가진다.

그림 27 - 장기보존패키지 생성 방식

7 변경이력 관리

7.1 개요

장기보존패키지는 최초 생성 이후 보존 과정에서 원문, 보존포맷 및 메타데이터 등의 변경이 발생할 수 있다. 이러한 변경 사항을 추적하고 기록하여

향후 원상복구 가능성과 신뢰성을 보장하기 위하여 변경이력 관리를 체계적으로 수행하여야 한다.

변경이력관리의 기본 원칙은 데이터 무결성과 중복 최소화로 한다. 각 버전의 변경 내역을 명확히 구분하여 관리함으로써 효율적이고 신뢰성 있는 장기보존체계를 구축하여야 한다. 데이터 무결성은 생성되거나 변환된 모든 장기보존패키지를 보존하고, 무결성 검증이 가능하도록 관리하여야 한다. 중복 최소화는 하나의 장기보존패키지 내 구성요소 중 중복 저장을 최소화하여 시스템 자원의 효율성을 높이는 원칙이다.

장기보존패키지는 이용의 직관성과 최신성 보장을 위해 항상 최신본을 패키지 루트에 유지하여야 한다. 루트에는 해당 시점의 효력을 가지는 메타데이터와 파일이 포함되어야 하며, 관리자는 별도의 탐색 없이 최신 상태의 패키지 구성과 내용을 확인할 수 있어야 한다. 이 원칙은 장기보존 환경에서 요구되는 열람·점검·검증 요청에 신속히 대응하기 위한 기본 구조로 기능한다.

변경이 발생한 경우, 패키지는 /Change History/vN/ 폴더를 생성하여 변경 직전의 파일을 원래 폴더 구조 그대로 보존하여야 한다. 버전번호 N은 변경이 발생할 때 마다 1씩 증가하며, 각 이력 폴더는 해당 시점의 복원에 필요한 최소 파일만 포함하여야 한다. 최신본은 루트에 항상 유지되어야 하며, 무결성 보장을 위해 Signature.xml과 Certificate.cer을 함께 보존하여야 한다. 또한 패키지의 전체 구성요소와 참조 관계를 확인하기 위하여 PackagingInformation.xml을 필수적으로 포함하여야 한다.

비고 1 vN을 새로운 버전, v(N-1)을 이전 버전으로 정의할 때, vN 버전은 변경된 구성요소를 포함하여 전체 구성요소를 유지한다. v(N-1) 버전은 변경된 구성요소의 직전 버전을 반드시 포함하여야 하며, 변경되지 않은 구성요소의 직전 버전은 필요에 따라 선택적으로 포함할 수 있다.

비고 2 대규모 업데이트가 이루어진 경우(예: 최초 생성 버전 또는 구성요소 대부분이 변경된 버전 등)에는 해당 버전의 모든 구성요소를 보존하는 것이 복원 시 명확한 기준점을 제공하며, 효율적인 복구를 가능하게 한다.

본 방안은 루트 기반 최신본 유지를 통해 접근성과 관리성을 확보하고, 변경 이전 파일만 보존하는 이력 구조로 저장 효율을 극대화한다. 전자서명과 패키징 정보의 상시 포함으로 무결성 검증을 체계화하고, 역방향 병합 복원으로 목표 시점의 패키지를 안정적으로 재현함으로써 장기보존 운용의 실효성과 신뢰성을 달성한다.

7.2 변경이력 관리 방안

표 2는 장기보존패키지 변경사항 발생 시 적용되는 변경이력 관리방안의 단계별 절차이다.

표 2 - 변경이력 관리방안의 단계별 절차

방안		설명
1	변경 감지	Metadata.xml, PackagingInformation.xml, Certificate.cer, Signature.xml, Original Content/ 폴더, Preservation Format/ 폴더 하위의 파일 중 하나라도 변경이 발생했는지 확인
2	버전 번호 증가 및 폴더 생성	기존/Change History/vN/중 가장 큰 버전 번호를 찾아 +1 한 폴더 /Change History/vN+1/을 생성
3	변경 전 파일 복사	변경되기 직전의 파일만 선택하여 /Change History/vN+1/하위에 원래 폴더 및 경로 구조 그대로 복사(필요한 경우 변경되지 않은 파일의 일부 또는 전부 보존가능) ※ Metadata.xml, PackagingInformation.xml, Certificate.cer, Signature.xml도 포함
4	변경된 최신본 반영	루트 디렉토리(/)에 변경된 Metadata.xml, Signature.xml, PackagingInformation.xml, Certificate.cer, 구성요소 파일등을 반영하여 최신 상태로 유지

장기보존패키지의 변경이력 관리 절차를 설명한 예시는 **그림 28**부터 **그림 32**이다. 예시는 7개의 구성요소로 이루어진 장기보존패키지를 기준으로 하며, 이후 1개 이상 구성요소가 변경됨에 따라 v2, v3, v4, v5 버전의 장기보존패키지가 순차적으로 생성되는 과정을 나타낸다. 이를 통해 버전 변경 시각 단계에서 변경이력이 어떻게 관리되는지 예시로 설명한다.

7개의 구성요소를 포함한 v1 버전의 장기보존패키지는 **그림 28**이다. v1의 구성요소들은 장기보존패키지의 '/'에 저장한다.

그림 28 - 변경이력 관리방안: v1 최초 생성

(3)요소가 변경되어 기존의 (3)요소만 v1에 유지하고 변경된 (3)요소를 포함한 v2를 생성한다. v2의 모든 구성요소는 장기보존패키지의 '/'에 저장하고, v1과 관련된 구성요소는 Change History/v1/ 경로에 보관한다.

그림 29 - 변경이력 관리방안: (3)요소 변경

(4)요소가 변경되어 v2의 (4)요소만 v2에 유지하고 변경된 (4)요소를 포함한 v3을 생성한다. v3의 모든 구성요소는 장기보존패키지 '/'에 저장하고, v2에 속하는 구성요소는 Change History/v2/ 경로에 보관한다.

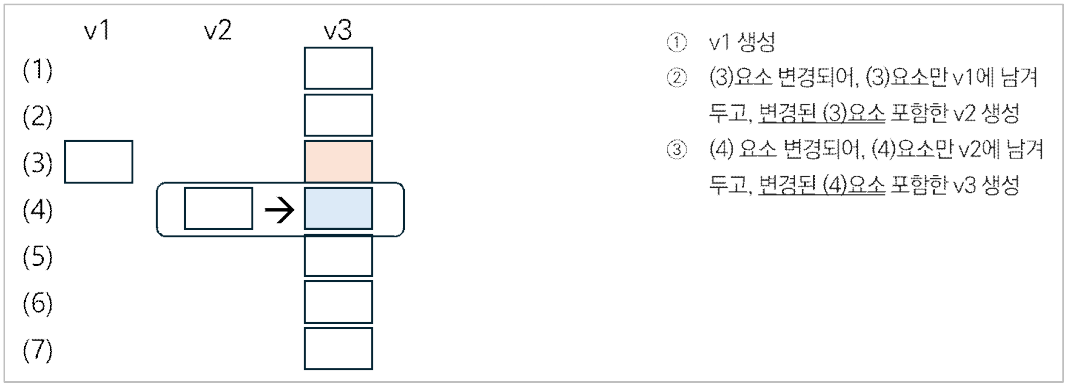


그림 30 - 변경이력 관리방안: (4)요소 변경

(1)요소와 (5)요소가 변경되어 v3의 (1)요소와 (5)요소만 v3에 유지하고 변경된 (1)요소와 (5)요소를 포함한 v4를 생성한다. v4의 모든 구성요소는 장기보존패키지 '/'에 저장하고, v3에 속하는 구성요소는 Change History/v3/ 경로에 보관한다.

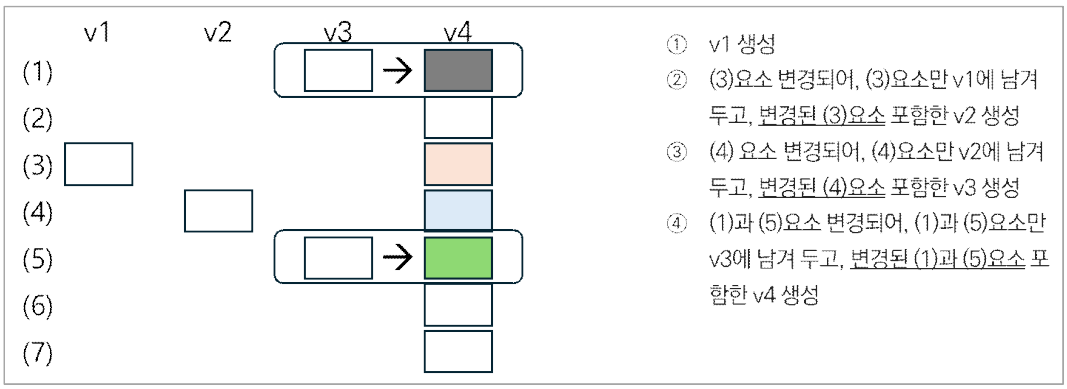


그림 31 - 변경이력 관리방안: (1)과 (5) 요소 변경

(5)요소가 변경되어 v4의 (5)요소만 v4에 유지하고 변경된 (5)요소를 포함한 v5를 생성한다. v5의 모든 구성요소는 장기보존패키지 '/'에 저장하고, v4에 속하는 구성요소는 Change History/v4/ 경로에 보관한다.

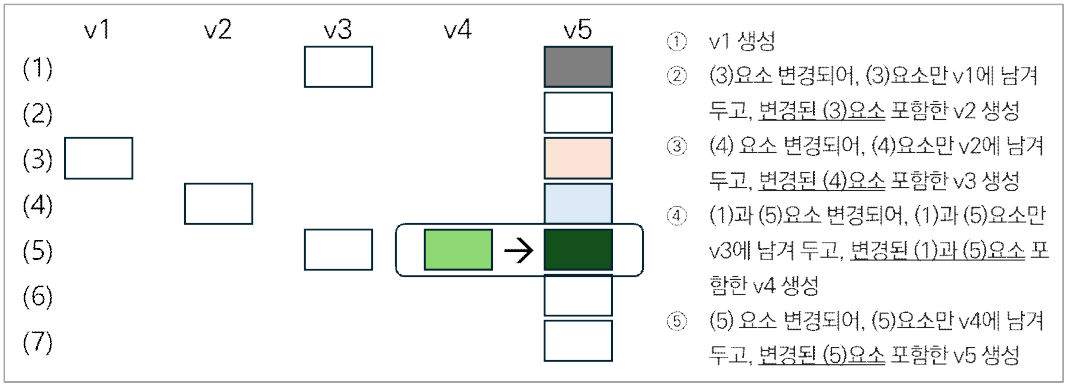


그림 32 - 변경이력 관리방안: (5) 요소 변경

7.3 변경이력 복원 방안

장기보존패키지의 복원은 역방향 병합 원리에 따라 수행하여야 한다. 복원 대상 버전을 v(p) (단, 1<p<n)으로 가정할 때, 복원은 최신본 v(n)에서 시작하여 v(n-1), v(n-2) ... v(p) 순서로 진행한다. 즉, v(n)의 구성요소를 v(n-1)에 덮어쓰고 동일한 과정을 v(p)까지 반복함으로써 복원을 완료한다.
복원절차의 개념적 구조는 **그림 33**와 같다. 변경이력 복원은 복원 대상 버전 이후 단계의 번경 이력을 역순으로 탐색하고 다시 변경되기 직전 버전의 구성요소를 덮어써 가는 방식으로 수행한다.

따라서 v2의 요소를 복원할 때 (1)요소는 v3의 (1)요소로, (2)요소는 v5의 (2)요소로, (3)요소는 v5의 (3)요소로, (4)요소는 v2의 (4)요소로, (5)요소는 v3의 (5)요소로, (6)요소는 v5의 (6)요소로, (7)요소는 v5의 (7)요소로 복원한다.
실제 복원 과정에서는 최신본 v(n)에서 출발하여 변경 이력을 역순으로 점검하고, 복원 대상 버전v(p)에서 다시 변경되기 전의 구성요소를 순차적으로 덮어써 나감으로써 해당 시점의 완전한 장기보존패키지를 복원하여야 한다.

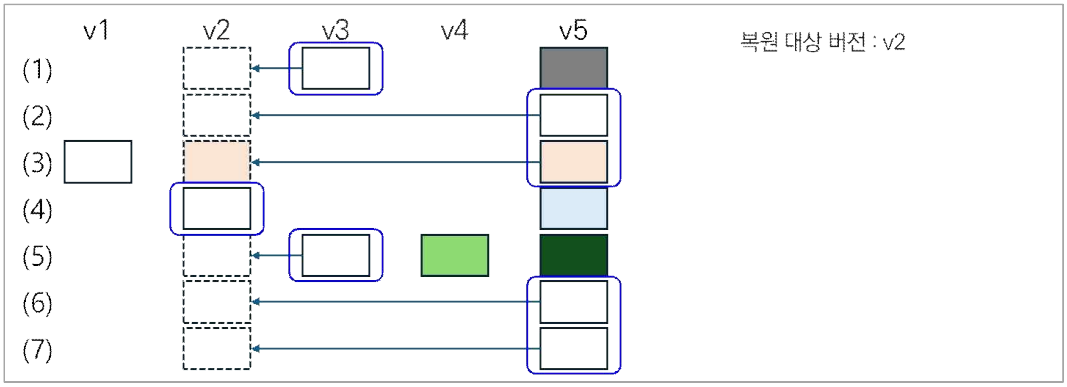


그림 33 - 변경이력 복원방안 원리에 대한 개념

표 3은 장기보존패키지 변경이력 복원 방안의 단계별 절차이다.

표 3 - 변경이력 복원방안의 단계별 절차

방안		설명
1	복원 대상 버전 지정	v1, v2, …, vN중복원하려는 장기보존패키지의 버전 번호(vP, 1≤P<N)를 선택한다.
2	최신본기준 복사	현재 루트(/)에 위치한 최신본전체를 특정 폴더로 복사한다. /restore_vP/라 하자. 이 복제본을 기반으로 복원을 진행한다.
3	패키징 정보 기반 역순 병합	/Change History/vN/부터 /Change History/vP/까지 순차적으로 역순 탐색하며, 해당 버전에 저장된 파일을 복제본에 복사한다. 파일 경로 구조는 원본과 동일하게 유지되어 있으므로 자동 병합이 가능하다. 이때, /Change History/vP/PackagingInformation.xml에 정의되지 않은 파일은 제외하고 복사한다.
4	구성요소 확인	/Change History/vP/PackagingInformation.xml에 정의된 파일들의 목록을 이용하여 /restore_vP/에 존재하는지 확인하고, 나머지 파일들을 삭제한다. vP 버전 생성 이후 보존 과정(예: 보존포맷 변환, 재현 도구 변경 등)에서 장기보존패키지 내에 추가 파일이 생성될 수 있다. 3단계 절차에서는 이러한 파일들이 vP와 함께 복사되므로, 복원 시에는 해당 파일을 장기보존패키지에서 제거해야 한다.
5	무결성 검증	/Change History/vP/PackagingInformation.xml에 정의된 해시값과 각 파일의 실제 해시를 비교하여 복원 결과의 무결성을 검증한다.

장기보존패키지의 변경이력 복원 절차를 설명한 예시는 **그림 34**부터 **그림 38**이다. 예시는 복원 대상을 v2 버전으로 설정하고 7개의 구성요소를 포함한 장기보존패키지가 v1 버전으로 최초 생성된다고 가정한다. 이후 1개 이상의 구성요소가 변경되어 v2, v3, v4, v5 버전의 장기보존패키지가 순차적으로

생성되는 과정을 통해 변경이력이 어떻게 관리되고 복원되는지 설명한다.

장기보존패키지 변경이력 복원 절차는 가장 최신본인 v5에서 시작한다. v5 전체를 '복원 v2'에 복사한다.

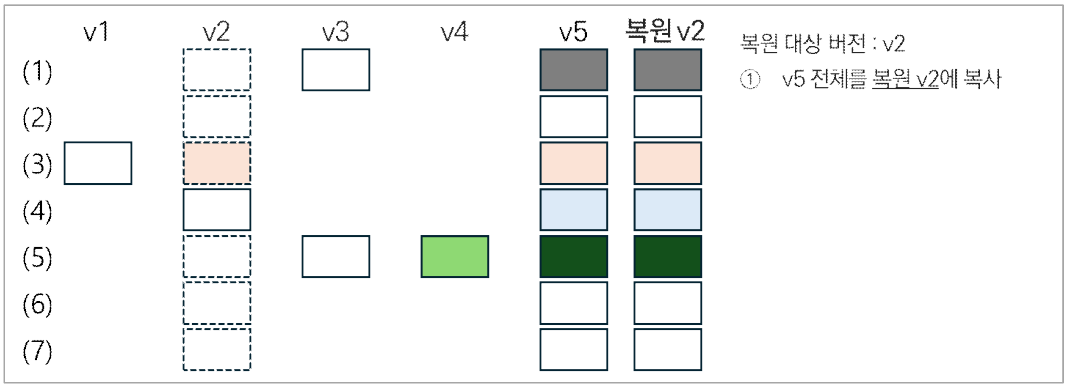


그림 34 - 변경이력 복원 방안: 최신본 v5 복사

v4의 내용으로 '복원 v2'의 해당 요소인 요소(5)를 덮어쓰기 한다.

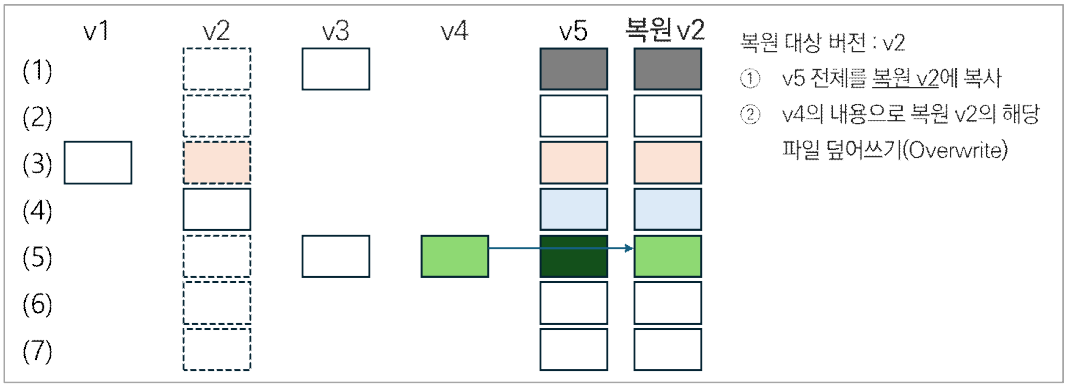


그림 35 - 변경이력 복원 방안: 최신본 v4 버전의 내용 복원

v3의 내용으로 '복원 v2'의 해당 요소인 요소(1)과 요소(5)를 덮어쓰기 한다.

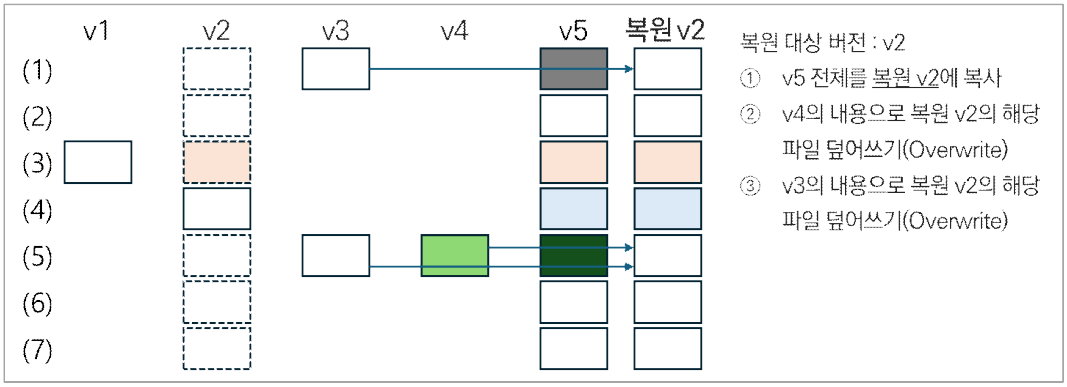


그림 36 - 변경이력 복원 방안: 최신본 v3 버전의 내용 복원

v2의 내용으로 '복원 v2'의 해당 요소인 요소(4)를 덮어쓰기 한다.

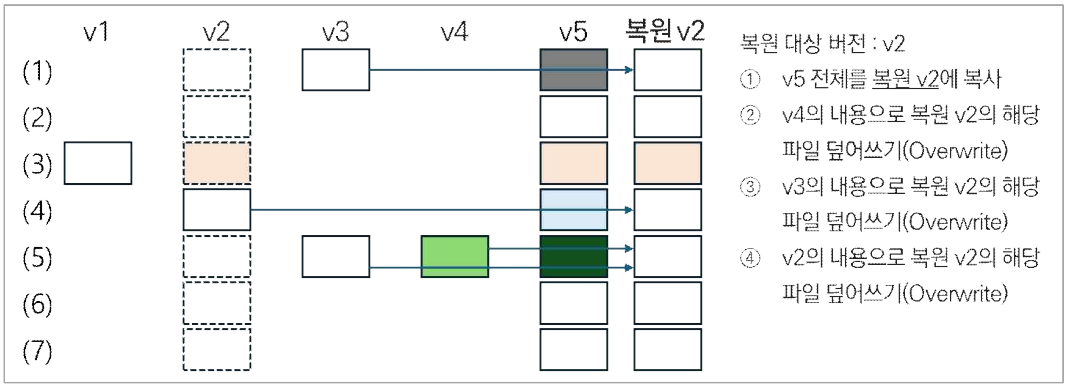


그림 37 - 변경이력 복원 방안: 최신본 v2 버전의 내용 복원

복원 대상인 v2 버전 장기보존패키지의 모든 내용이 복원 완료되었다.

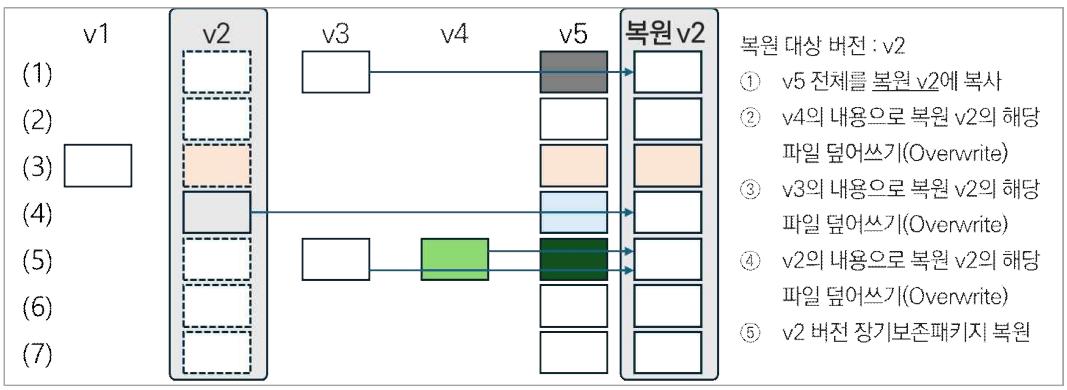


그림 38 - 변경이력 복원 방안: '복원 v2' 복원 완료

부속서 A(참고)
NEO3 장기보존패키지 구성 사례

<?xml version="1.0" encoding="UTF-8"?> <xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" targetNamespace="https://www.archives.go.kr/neo3" xmlns="https://www.archives.go.kr/neo3" elementFormDefault="qualified" attributeFormDefault="unqualified" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:vc="http://www.w3.org/2007/XMLSchema-versioning" vc:minVersion="1.1" xmlns:dc="http://purl.org/dc/elements/1.1/" xmlns:ead="http://ead3.archivists.org/schema/"> <xsd:import namespace="http://www.w3.org/XML/1998/namespace" schemaLocation="http://www.w3.org/2001/03/xml.xsd"/> <xsd:import namespace="http://purl.org/dc/elements/1.1/" schemaLocation="http://dublincore.org/schemas/xmls/simpledc20021212.xsd"/> <xsd:import namespace="http://ead3.archivists.org/schema/" schemaLocation="https://www.loc.gov/ead/ead3.xsd"/ <xsd:element name="NEO3Metadata"> <xsd:complexType> <xsd:sequence minOccurs="1"> <xsd:element ref="PreservationDescInfo"/> <xsd:element ref="DescriptiveInformation" minOccurs="0"/> <xsd:element ref="ContentInformation" minOccurs="0"/>	① NEO3 장기보존 패키지의 최상위 메타데이터 요소. 하위에는 장기보존메타데이터, 기술 정보, 콘텐트 정

</xsd:sequence> <xsd:attribute name="KDN" type="xsd:string" default="NEO3 최상위 요소"/> </xsd:complexType> </xsd:element> <xsd:element name="PreservationDescInfo"> <xsd:annotation> <xsd:documentation>XML에서 PreservationDescInfo 요소의 "type" attribute가 'dataset'로 설정되면 Type_PreservationDescInfoForDataset가 "document" 로 선언되면, Type_PreservationDescInfoForDocument가 설정된다.</xsd:documentation> </xsd:annotation> <xsd:alternative test="@type = 'dataset'" type="Type_PreservationDescInfoForDataset"/> <xsd:alternative test="@type = 'document'" type="Type_PreservationDescInfoForDocument"/> </xsd:element> <xsd:element name="DescriptiveInformation"> <xsd:annotation> <xsd:documentation>NEO3 장기보존 패키지 기술 정보(Descriptive Information)</xsd:docu mentation> </xsd:annotation> <xsd:complexType> <xsd:sequence> <xsd:element name="DescriptiveMetadata" minOccurs="0" maxOccurs="unbounded"> <xsd:alternative test="@scheme = 'DC' and @type = '데이터세트_전체정보'" type="TypeDC_TotalDatasetInfo"/> <xsd:alternative test="@scheme = 'DC' and @type = '데이터세트_단위정보'" type="TypeDC_SingleDatasetInfo"/> <xsd:alternative test="@scheme = 'DC' and not(@type)" type="TypeDC_Default"/>	보를 포함. ②PreservationDescInfo 요소는 패키지의 장기보존 메타데이터를 정의함. 이 요소의 "type" 속성 값에 따라 문서형(document) 또는 데이터세트형(dataset)으로 달라짐. ③ DescriptiveInformation 요소는 기술 정보로 DC/EAD 스키마를 지원함.

<xsd:alternative test="@scheme = 'EAD' and @type = '데이터세트_전체정보'" type="TypeEAD_TotalDatasetInfo"/> <xsd:alternative test="@scheme = 'EAD' and @type = '데이터세트_단위정보'" type="TypeEAD_SingleDatasetInfo"/> <xsd:alternative test="@scheme = 'EAD' and not(@type)" type="TypeEAD_Default"/> </xsd:element> </xsd:sequence> <xsd:attribute default="기술정보(Descriptive Information)" name="KDN" type="xsd:string"/> </xsd:complexType> </xsd:element> <xsd:element name="ContentInformation"> <xsd:annotation> <xsd:documentation>NEO3 장기보존 패키지 원문 및 보존포맷 콘텐츠 정보(Content Information)</xsd:documentation> </xsd:annotation> <xsd:complexType> <xsd:sequence minOccurs="1"> <xsd:element maxOccurs="unbounded" minOccurs="0" name="ContentItem" type="TypeContentItem"> </xsd:element> </xsd:sequence> <xsd:attribute name="KDN" type="xsd:string" default="콘텐츠정보(Content Information)"/> </xsd:complexType> </xsd:element> <xsd:complexType name="Type_PreservationDescInfoForDocument"> <xsd:annotation>	④ ContentInformation 요소는 콘텐트 정보(원문 및 보존포맷 파일)를 정의함. ⑤ ContentItem 요소는 하나의 파일 또는 객체 단위를 나타냄. 반복이 가능하므로(unbounded) 여러 개의 콘텐트를 넣을 수 있음.

<xsd:documentation> [문서유형] 장기보존패키지 기록물건,기록물철 객체 스키마 장기보존메타데이터</xsd:documentation> \</xsd:annotation> <xsd:sequence> <xsd:element name="ObjectMetadata" minOccurs="1"> <xsd:annotation> <xsd:documentation>객체 공통 메타데이터(기록물철, 기록물건 공통 메타데이터)</xsd:documentation> \</xsd:annotation> <xsd:complexType> <xsd:sequence minOccurs="1"> <xsd:element name="ObjectType" minOccurs="1"> <xsd:complexType> <xsd:simpleContent> <xsd:extension base="EnumObjectType"> <xsd:attribute name="KDN" type="xsd:string" default="객체유형"/> \</xsd:extension> \</xsd:simpleContent> \</xsd:complexType> \</xsd:element> <xsd:element name="ObjectTypeDescription" minOccurs="1"> <xsd:complexType> <xsd:simpleContent> <xsd:extension base="xsd:string"> <xsd:attribute name="KDN" type="xsd:string" default="객체유형설명"	⑥ [문서형] 장기보존 패키지의 기록물건·철 객체 스키마에 대한 장기보존 메타데이터를 정의함. ⑦ 객체 공통 메타데이터(ObjectMetadata)는 기록물철·건의 문서유형 객체에 공통적으로 적용되는 속성을 정의함.

```
                    />
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ObjectCreationDate" minOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:dateTime">
                    <xsd:attribute name="KDN" type="xsd:string" default="객체생성일시"
                    />
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ObjectSubType" minOccurs="0">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="EnumObjectSubType">
                    <xsd:attribute default="객체상세유형" name="KDN" type="xsd:string"
                    />
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
```

</xsd:sequence> <xsd:attribute name="KDN" type="xsd:string" default="객체 메타데이터"/> </xsd:complexType> </xsd:element> <xsd:element name="ObjectContent" minOccurs="1"> <xsd:annotation> <xsd:documentation>객체 콘텐츠 메타데이터(기록물철, 기록물건)</xsd:documentation> </xsd:annotation> <xsd:complexType> <xsd:choice> <xsd:element ref="File"/> <xsd:element ref="Record"/> </xsd:choice> <xsd:attribute name="KDN" type="xsd:string" default="객체 콘텐츠"/> </xsd:complexType> </xsd:element> <xsd:element name="Components" minOccurs="0"> <xsd:annotation> <xsd:documentation>컴포넌트 목록 정보</xsd:documentation> </xsd:annotation> <xsd:complexType> <xsd:sequence> <xsd:element maxOccurs="unbounded" minOccurs="0" name="Component" type="TypeComponent"> </xsd:element> </xsd:sequence>	⑧ 객체 콘텐트 메타데이터(ObjectContent)는 File 또는 Record 요소 중 하나를 선택하여 객체의 실제 내용을 구조적으로 표현함. ⑨ 컴포넌트 목록(Components)은 객체(ObjectContent)를 구성하는 요소들의 목록 정보를 정의함.

<xsd:attribute name="KDN" type="xsd:string" default="컴포넌트목록"/> </xsd:complexType> </xsd:element> </xsd:sequence> <xsd:attribute name="KDN" type="xsd:string" default="장기보존 메타데이터"/> </xsd:complexType> <xsd:element name="Record"> <xsd:annotation> <xsd:documentation>건 기록물</xsd:documentation> </xsd:annotation> <xsd:complexType> <xsd:sequence> <xsd:element name="RecordMetadata"> <xsd:complexType> <xsd:sequence> <xsd:element name="AgentCon" type="TypeAgent" maxOccurs="unbounded"/> <xsd:element name="CreatorCon" type="TypeCreater"> </xsd:element> <xsd:element name="RecordIdentifierCon" type="TypeRecordIdentifier"/> <xsd:element name="TitleCon" type="TypeTitle"/> <xsd:element maxOccurs="unbounded" minOccurs="0" name="DescriptionCon" type="TypeDescription"/> <xsd:element name="SubjectCon" type="TypeSubject" maxOccurs="unbounded" minOccurs="0"/> <xsd:element name="IsDigitalRecord"> <xsd:complexType>	⑩ Record 요소는 '건 기록물'로 건 단위의 메타데이터를 기술함.

```
            <xsd:simpleContent>
                <xsd:extension base="EnumIsDigitalRecordType">
                    <xsd:attribute name="KDN" default="전자기록물여부"
                        type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="StorageCon" type="TypeRecordStorage"/>
    <xsd:element name="ClassificationInfo" type="TypeClassification"
        minOccurs="1" maxOccurs="1"/>
    <xsd:element name="CreationHistoryCon" type="TypeCreationHistory"/>
    <xsd:element name="RetentionPeriodCon" type="TypeRetentionPeriod"
        minOccurs="0"/>
    <xsd:element name="RightsCon" type="TypeRights"/>
    <xsd:element maxOccurs="unbounded" minOccurs="0"
        name="ManagementHistoryCon" type="TypeManagementHistory"/>
    <xsd:element maxOccurs="unbounded" minOccurs="0" name="UseHistoryCon"
        type="TypeUseHistory"/>
    <xsd:element maxOccurs="unbounded" minOccurs="0"
        name="PreservationHistoryCon" type="TypePreservationHistory"/>
    <xsd:element maxOccurs="unbounded" minOccurs="0" name="RelationCon"
        type="TypeRelation"/>
    <xsd:element maxOccurs="1" minOccurs="0" name="DestructionFreezeCon"
        type="TypeDestructionFreeze">
```

⑪

<xsd:annotation> <xsd:documentation>기록물 폐기금지</xsd:documentation> </xsd:annotation> </xsd:element> <xsd:element maxOccurs="unbounded" minOccurs="0" name="IntegrityCheck" type="TypeIntegrityCheck"> </xsd:element> <xsd:element maxOccurs="unbounded" minOccurs="0" name="CorrectionCon"> <xsd:annotation> <xsd:documentation>기록물 부기정정</xsd:documentation> </xsd:annotation> <xsd:complexType> <xsd:sequence> <xsd:element name="CorrectionType"> <xsd:complexType> <xsd:simpleContent> <xsd:extension base="EnumCorrectionType"> <xsd:attribute fixed="부기정정유형" name="KDN" type="xsd:string"/> </xsd:extension> </xsd:simpleContent> </xsd:complexType> </xsd:element> <xsd:element name="CorrectionNumber"> <xsd:complexType> <xsd:simpleContent>	DestructionFreezeCon 요소는 해당 기록물의 폐기를 금지하거나 중지하는 상태를 표시함. ⑫ CorrectionCon 요소는 기록물의 부기정정 내역을 기술함.

```
                <xsd:extension base="xsd:string">
                <xsd:attribute fixed="부기정정접수번호" name="KDN"
                type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="CorrectionRequesterDateTime">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:dateTime">
                <xsd:attribute fixed="부기정정요청일자" name="KDN"
                type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="1" name="CorrectionRequesterIDRef">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:IDREF">
                <xsd:attribute fixed="부기정정요청자" name="KDN"
                type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
```

```
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="CorrectionReasons">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                <xsd:attribute fixed="부기정정사유" name="KDN"
                type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="CorrectionReceptDateTime">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:dateTime">
                <xsd:attribute fixed="부기정정접수일자" name="KDN"
                type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="1"
        name="CorrectionReceptionistIDRef">
        <xsd:complexType>
```

```
            <xsd:simpleContent>
              <xsd:extension base="xsd:IDREF">
              <xsd:attribute fixed="부기정정접수자" name="KDN"
              type="xsd:string"/>
              </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="CorrectionImplementDateTime">
        <xsd:complexType>
            <xsd:simpleContent>
              <xsd:extension base="xsd:dateTime">
              <xsd:attribute fixed="부기정정처리일자" name="KDN"
              type="xsd:string"/>
              </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="CorrectionDescription">
        <xsd:complexType>
            <xsd:simpleContent>
              <xsd:extension base="xsd:string">
              <xsd:attribute fixed="부기정정설명" name="KDN"
              type="xsd:string"/>
              </xsd:extension>
```

```
                                    </xsd:simpleContent>
                                </xsd:complexType>
                            </xsd:element>
                            <xsd:element minOccurs="0" name="CorrectionFilename">
                                <xsd:complexType>
                                    <xsd:simpleContent>
                                      <xsd:extension base="xsd:string">
                                      <xsd:attribute fixed="부기정정대상파일" name="KDN"
                                      type="xsd:string"/>
                                      </xsd:extension>
                                    </xsd:simpleContent>
                                </xsd:complexType>
                            </xsd:element>
                        </xsd:sequence>
                        <xsd:attribute fixed="부기정정이력" name="KDN" type="xsd:string"/>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
            <xsd:attribute name="KDN" type="xsd:string" default="기록건공통메타데이터"/>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ExtraMetadata" minOccurs="0" type="TypeRecordSub"> </xsd:element>
</xsd:sequence>
<xsd:attribute name="KDN" type="xsd:string" default="기록건"/>
</xsd:complexType>
```

```
</xsd:element>
<xsd:element name="File">
    <xsd:annotation>
        <xsd:documentation>철 기록물</xsd:documentation>
    </xsd:annotation>
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="FileMetadata">
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element name="AgentCon" type="TypeAgent" maxOccurs="unbounded"/>
                        <xsd:element name="CreatorCon" type="TypeCreater"> </xsd:element>
                        <xsd:element name="RecordIdentifierCon" type="TypeRecordIdentifier"/>
                        <xsd:element name="TitleCon" type="TypeTitle"/>
                        <xsd:element maxOccurs="unbounded" minOccurs="0" name="DescriptionCon"
                            type="TypeDescription"/>
                        <xsd:element name="SubjectCon" type="TypeSubject" maxOccurs="unbounded"
                            minOccurs="0"/>
                        <xsd:element name="IsDigitalRecord">
                            <xsd:complexType>
                                <xsd:simpleContent>
                                    <xsd:extension base="EnumIsDigitalRecordType">
                                        <xsd:attribute name="KDN" default="비전자여부"
                                            type="xsd:string"/>
                                    </xsd:extension>
```

⑬ File은 '기록물 철'로, 철 단위의 메타데이터를 기술함.

```
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="StorageCon" type="TypeRecordStorage"/>
    <xsd:element name="ClassificationInfo" type="TypeClassification"
        minOccurs="1" maxOccurs="1"/>
    <xsd:element name="CreationHistoryCon" type="TypeCreationHistory"
        minOccurs="0"/>
    <xsd:element name="RetentionPeriodCon" type="TypeRetentionPeriod"/>
    <xsd:element name="PreservationPlace">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="EnumPreservationPlaceType">
                    <xsd:attribute default="보존장소" name="KDN"
                        type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="RightsCon" type="TypeRights" minOccurs="0"/>
    <xsd:element maxOccurs="unbounded" minOccurs="0"
        name="ManagementHistoryCon" type="TypeManagementHistory"/>
    <xsd:element maxOccurs="unbounded" minOccurs="0" name="UseHistoryCon"
        type="TypeUseHistory"/>
    <xsd:element maxOccurs="unbounded" minOccurs="0"
```

```
    name="PreservationHistoryCon" type="TypePreservationHistory"/>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="RelationCon"
    type="TypeRelation"/>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="ResultCon"
    type="TypeResult"/>
<xsd:element maxOccurs="unbounded" minOccurs="0" name="WorkHandBooksCon"
    type="TypeWorkHandBooks"/>
<xsd:element minOccurs="0" name="FileCV">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="EnumFileCVType">
                <xsd:attribute name="KDN" default="과제유형"
                    type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
<xsd:element minOccurs="0" name="FileRole">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="EnumFileRoleType">
                <xsd:attribute name="KDN" default="과제역할구분"
                    type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
```

</xsd:complexType> </xsd:element> <xsd:element maxOccurs="1" minOccurs="0" name="DestructionFreezeCon" type="TypeDestructionFreeze"> <xsd:annotation> <xsd:documentation>기록물 폐기금지</xsd:documentation> </xsd:annotation> </xsd:element> <xsd:element maxOccurs="unbounded" minOccurs="0" name="IntegrityCheck" type="TypeIntegrityCheck"> </xsd:element> </xsd:sequence> <xsd:attribute name="KDN" type="xsd:string" default="기록물철메타데이터"/> </xsd:complexType> </xsd:element> <xsd:element name="ExtraMetadata" minOccurs="0" type="TypeRecordSub"> </xsd:element> </xsd:sequence> <xsd:attribute name="KDN" type="xsd:string" default="기록물철"/> </xsd:complexType> </xsd:element> <xsd:simpleType name="EnumHashAlgorithmType"> <xsd:restriction base="xsd:string"> <xsd:enumeration value="CBC"/> <xsd:enumeration value="HMAC"/> <xsd:enumeration value="MD2"/> <xsd:enumeration value="MD4"/>	⑭ DestructionFreezeCon 요소는 해당 기록물의 폐기를 금지하거나 중지하는 상태를 표시함. ⑮ ExtraMetadata 요소는 기록물철에 부가적으로 필요한 확장 메타데이터를 정의함.

```
            <xsd:enumeration value="MD5"/>
            <xsd:enumeration value="MAC"/>
            <xsd:enumeration value="SHA1"/>
            <xsd:enumeration value="SHA2-224"/>
            <xsd:enumeration value="SHA2-256"/>
            <xsd:enumeration value="SHA2-384"/>
            <xsd:enumeration value="SHA2-512"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumObjectType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="기록철"/>
            <xsd:enumeration value="기록물철"/>
            <xsd:enumeration value="기록건"/>
            <xsd:enumeration value="기록물건"/>
            <xsd:enumeration value="수정된기록철"/>
            <xsd:enumeration value="수정된기록물철"/>
            <xsd:enumeration value="수정된기록건"/>
            <xsd:enumeration value="수정된기록물건"/>
            <xsd:enumeration value="컴포넌트"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumObjectSubType">
        <xsd:restriction base="xsd:string">
```

```
            <xsd:enumeration value="문서보고"/>
            <xsd:enumeration value="메모보고"/>
            <xsd:enumeration value="지시사항"/>
            <xsd:enumeration value="전자심의"/>
            <xsd:enumeration value="기타"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:complexType name="TypeComponent">
        <xsd:sequence>
            <xsd:element name="ComponentTitle">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="컴포넌트제목" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="ComponentType">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="EnumComponentType">
                            <xsd:attribute default="컴포넌트유형" name="KDN" type="xsd:string"/>
                        </xsd:extension>
```

</xsd:simpleContent> </xsd:complexType> </xsd:element> <xsd:element name="ComponentPath"> <xsd:annotation> <xsd:documentation>패키지 내부 루트 기준 상대경로</xsd:documentation> </xsd:annotation> <xsd:complexType> <xsd:simpleContent> <xsd:extension base="xsd:string"> <xsd:attribute default="컴포넌트경로" name="KDN" type="xsd:string"/> </xsd:extension> </xsd:simpleContent> </xsd:complexType> </xsd:element> <xsd:element name="ComponentOrder" minOccurs="0"> <xsd:complexType> <xsd:simpleContent> <xsd:extension base="xsd:string"> <xsd:attribute default="컴포넌트순서" name="KDN" type="xsd:string"/> </xsd:extension> </xsd:simpleContent> </xsd:complexType> </xsd:element> <xsd:element name="ComponentSize">	⑯ ComponentPath 요소는 패키지 내부에서 해당 컴포넌트 파일이 위치한 상대경로를 기술함

```
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="컴포넌트크기" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="0" name="ComponentVersion">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="컴포넌트버전" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="0" name="ComponentHash">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="컴포넌트해쉬" name="KDN" type="xsd:string"/>
                    <xsd:attribute default="SHA2-512" name="HashAlgorithm"
                        type="EnumHashAlgorithmType"/>
                </xsd:extension>
```

```
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element maxOccurs="unbounded" minOccurs="0" name="ComponentPreservationHistoryCon">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="ComponentPreservationType">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="EnumComponentPreservationType">
                                <xsd:attribute default="컴포넌트 보존처리유형" name="KDN"
                                    type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="ComponentPreservationDescription">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="컴포넌트 보존처리 설명" name="KDN"
                                    type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
```

```
                </xsd:element>
                <xsd:element name="ComponentPreservationDate">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:dateTime">
                                <xsd:attribute default="컴포넌트 보존처리 일시" name="KDN"
                                    type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="ComponentPreserveAgentIDRef" minOccurs="0">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:IDREF">
                                <xsd:attribute default="컴포넌트 보존처리행위자" name="KDN"
                                    type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
            <xsd:attribute default="컴포넌트 보존이력" name="KDN" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>
```

```
<xsd:element maxOccurs="unbounded" minOccurs="0" name="ComponentRelationIDRef">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:IDREF">
                <xsd:attribute default="컴포넌트 관계정보 ID참조" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
<xsd:element minOccurs="0" name="IsCopy">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:boolean">
                <xsd:attribute default="복사본여부" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
<xsd:element minOccurs="0" name="OriginalFileID">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute default="생산시스템 파일관리ID" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
```

```
            </xsd:complexType>
        </xsd:element>
        <xsd:element minOccurs="0" name="OriginalFilename">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="원본파일명" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute name="ComponentID" type="xsd:string"/>
    <xsd:attribute default="컴포넌트" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="TypeAgent">
    <xsd:sequence>
        <xsd:element name="AgentType">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="EnumAgentType">
                        <xsd:attribute default="행위자유형" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
```

```
</xsd:element>
<xsd:element minOccurs="0" name="CorporateInfo">
    <xsd:complexType>
        <xsd:complexContent>
            <xsd:extension base="TypeCorporateInfo">
                <xsd:attribute default="소속기관정보" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>
</xsd:element>
<xsd:element minOccurs="0" name="SectionInfo">
    <xsd:complexType>
        <xsd:complexContent>
            <xsd:extension base="TypeSectionInfo">
                <xsd:attribute default="소속부서정보" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:complexContent>
    </xsd:complexType>
</xsd:element>
<xsd:element minOccurs="0" name="PersonalInfo">
    <xsd:complexType>
        <xsd:complexContent>
            <xsd:extension base="TypePersonalInfo">
                <xsd:attribute default="개인정보" name="KDN" type="xsd:string"/>
            </xsd:extension>
```

```
                    </xsd:complexContent>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute name="AgentID" type="xsd:ID"/>
        <xsd:attribute default="행위자" name="KDN" type="xsd:string"/>
    </xsd:complexType>
    <xsd:complexType name="TypeRecordIdentifier">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="GroupID">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="기록물군식별자" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element minOccurs="0" name="SubGroupID">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="기록물하위군식별자" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
```

```
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="0" name="SeriesID">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="기록물계열식별자" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="0" name="SubSeriesID">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="기록물하위계열식별자" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="FileID" minOccurs="0">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="기록물철고유ID" name="KDN" type="xsd:string"/>
```

```
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="RecordID" minOccurs="0">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="기록물건고유ID" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element maxOccurs="unbounded" minOccurs="0" name="SystemID">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="SystemIDType">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="EnumSystemIDType">
                                <xsd:attribute default="시스템식별자유형" name="KDN"
                                    type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
```

```
                </xsd:element>
                <xsd:element name="SystemIDValue">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="시스템식별자" name="KDN" type="xsd:string"
                                />
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
            <xsd:attribute default="시스템식별자 정보" name="KDN" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>
    <xsd:element maxOccurs="unbounded" minOccurs="0" name="SecondaryID">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="SecondaryIDType">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="EnumSecondaryIDType">
                                <xsd:attribute default="보조식별자유형" name="KDN"
                                    type="xsd:string"/>
                            </xsd:extension>
```

```
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="SecondaryIDValue">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="보조식별자" name="KDN" type="xsd:string"
                                />
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
            <xsd:attribute default="보조식별자 정보" name="KDN" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>
</xsd:sequence>
<xsd:attribute default="고유식별자" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="TypeTitle">
    <xsd:sequence>
        <xsd:element name="MainTitle">
            <xsd:complexType>
                <xsd:simpleContent>
```

```
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="제목" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element maxOccurs="unbounded" minOccurs="0" name="AlternativeTitleCon">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="AlternativeTitleType">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="EnumAlternativeTitleType">
                                <xsd:attribute default="기타제목유형" name="KDN" type="xsd:string"
                                />
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="AlternativeTitleWords">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="기타제목명" name="KDN" type="xsd:string"
                                />
```

```
                                    </xsd:extension>
                                </xsd:simpleContent>
                            </xsd:complexType>
                        </xsd:element>
                    </xsd:sequence>
                    <xsd:attribute default="기타제목" name="KDN" type="xsd:string"/>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute default="표제(제목)" name="KDN" type="xsd:string"/>
    </xsd:complexType>
    <xsd:complexType name="TypeDescription">
        <xsd:sequence>
            <xsd:element name="DescriptionType">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="EnumDescriptionType">
                            <xsd:attribute default="기술정보유형" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="DescriptionText">
                <xsd:complexType>
                    <xsd:simpleContent>
```

```
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="기술정보내용" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute default="기술정보" name="KDN" type="xsd:string"/>
    </xsd:complexType>
    <xsd:simpleType name="EnumIsDigitalRecordType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="전자기록물"/>
            <xsd:enumeration value="비전자기록물"/>
            <xsd:enumeration value="혼합기록물"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumSubjectType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="일반주제명"/>
            <xsd:enumeration value="인명"/>
            <xsd:enumeration value="단체명"/>
            <xsd:enumeration value="지명"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
```

```
</xsd:simpleType>
<xsd:complexType name="TypeSubject">
    <xsd:sequence>
        <xsd:element name="SubjectType">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="EnumSubjectType">
                        <xsd:attribute default="주제유형" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="SubjectWords" maxOccurs="unbounded">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="주제명" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="주제" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:simpleType name="EnumTypeOfCopy">
```

```
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="대체사본"/>
            <xsd:enumeration value="보존사본"/>
            <xsd:enumeration value="원본대체사본"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumAlternativeType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="문서관리카드"/>
            <xsd:enumeration value="메모보고"/>
            <xsd:enumeration value="지시사항"/>
            <xsd:enumeration value="전자심의"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumMediumType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="자기테이프"/>
            <xsd:enumeration value="자기디스크"/>
            <xsd:enumeration value="광매체"/>
            <xsd:enumeration value="반도체저장매체"/>
            <xsd:enumeration value="음반"/>
            <xsd:enumeration value="SSD"/>
            <xsd:enumeration value="파일"/>
```

```
            <xsd:enumeration value="스토리지"/>
            <xsd:enumeration value="종이"/>
            <xsd:enumeration value="필름"/>
            <xsd:enumeration value="플라스틱디스크"/>
            <xsd:enumeration value="인화지"/>
            <xsd:enumeration value="기타"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumExtentType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="기록물첨부수"/>
            <xsd:enumeration value="전자기록용량"/>
            <xsd:enumeration value="비전자기록용량"/>
            <xsd:enumeration value="전자파일개수"/>
            <xsd:enumeration value="보존포맷변환개수"/>
            <xsd:enumeration value="기록물건수"/>
            <xsd:enumeration value="기록물건포맷변환개수"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumUnitType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="B"/>
            <xsd:enumeration value="장"/>
```

```
            <xsd:enumeration value="쪽"/>
            <xsd:enumeration value="건"/>
            <xsd:enumeration value="점"/>
            <xsd:enumeration value="매"/>
            <xsd:enumeration value="기타"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumStructureType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="기록물분철"/>
            <xsd:enumeration value="기록물건"/>
            <xsd:enumeration value="기록물철"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumClassificationLevelType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="정책분야"/>
            <xsd:enumeration value="정책영역"/>
            <xsd:enumeration value="대기능"/>
            <xsd:enumeration value="중기능"/>
            <xsd:enumeration value="소기능"/>
            <xsd:enumeration value="단위과제"/>
            <xsd:enumeration value="단위업무"/>
```

```
            <xsd:enumeration value="세분류(단위업무)"/>
            <xsd:enumeration value="기록물철"/>
            <xsd:enumeration value="1차분류"/>
            <xsd:enumeration value="2차분류"/>
            <xsd:enumeration value="3차분류"/>
            <xsd:enumeration value="4차분류"/>
            <xsd:enumeration value="5차분류"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:complexType name="TypeCreationHistory">
        <xsd:sequence minOccurs="0">
            <xsd:element name="RecordCreationSystem" minOccurs="0">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="EnumRecordCreationSystemType">
                            <xsd:attribute default="생산시스템" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="CreationSectionCon" minOccurs="0">
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element name="CreationSectionIDRef">
```

```
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:IDREF">
                                <xsd:attribute default="생산부서(관계자) 참조ID" name="KDN"
                                    type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
            <xsd:attribute default="생산부서" name="KDN" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="CreationType" minOccurs="0">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="EnumCreationType">
                    <xsd:attribute default="생산유형" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element maxOccurs="unbounded" minOccurs="0" name="ReportPathCon">
        <xsd:complexType>
            <xsd:sequence minOccurs="0">
```

```
<xsd:element minOccurs="1" name="ReportPathOrder">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute fixed="보고순서" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
<xsd:element name="ReportPathAgentIDRef">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:IDREF">
                <xsd:attribute default="보고행위자" name="KDN" type="xsd:string"
                />
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
<xsd:element name="ReportPathType">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute default="보고유형" name="KDN" type="xsd:string"/>
            </xsd:extension>
```

```
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ReportPathOpinion" minOccurs="0">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="의견" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="0" name="ReportPathUniqueID">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute fixed="고유ID" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="0" name="ReportPathParentID">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
```

```
                        <xsd:attribute fixed="부모ID" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element minOccurs="0" name="ReportPathDepth">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute fixed="의견깊이" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element minOccurs="1" name="ComponentTypeCreated">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="EnumComponentTypeCreated">
                        <xsd:attribute fixed="생산컴포넌트유형" name="KDN"
type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element minOccurs="0" name="ComponentVersionCreated">
```

```
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute fixed="생산컴포넌트버전" name="KDN"
type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="ReportPathRequest">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="EnumReportRequestType">
                                <xsd:attribute default="요청상태" name="KDN" type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="ReportPathResult">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="EnumReportResultType">
                                <xsd:attribute default="처리결과" name="KDN" type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
```

```
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="ReportPathDate">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:dateTime">
                                <xsd:attribute default="처리일시" name="KDN" type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element minOccurs="0" name="OriginalFileID" maxOccurs="unbounded">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute fixed="생산시스템 파일관리ID" name="KDN"
                                    type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
            <xsd:attribute default="보고경로" name="KDN" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>
```

```
<xsd:element name="CreationDateTimeCon" minOccurs="0">
    <xsd:complexType>
        <xsd:sequence minOccurs="0">
            <xsd:element name="DateTimeCreated">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:dateTime">
                            <xsd:attribute default="생산일시" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="DateTimeClosed">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:dateTime">
                            <xsd:attribute default="종료일시" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="DateTimeRegistered" minOccurs="0">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:dateTime">
```

```
                    <xsd:attribute default="등록일시" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="DateTimeEnforced" minOccurs="0">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:dateTime">
                    <xsd:attribute default="시행일시" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="DateTimePropelled" minOccurs="0">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:dateTime">
                    <xsd:attribute default="추진일시" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ScheduledDateTimeStart" minOccurs="0">
        <xsd:complexType>
```

```
            <xsd:simpleContent>
                <xsd:extension base="xsd:dateTime">
                    <xsd:attribute default="시작예정일시" name="KDN" type="xsd:string"
                    />
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ScheduledDateTimeEnd" minOccurs="0">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:dateTime">
                    <xsd:attribute default="종료예정일시" name="KDN" type="xsd:string"
                    />
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="DateTimeDelivered" minOccurs="0">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:dateTime">
                    <xsd:attribute default="발송일자" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
```

```
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="DateTimeOrdered" minOccurs="0">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:dateTime">
                                <xsd:attribute default="지시일자" name="KDN" type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="DateTimeReceipt" minOccurs="0">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:dateTime">
                                <xsd:attribute default="접수일자" name="KDN" type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
            <xsd:attribute default="생산이력" name="KDN" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>
</xsd:sequence>
```

```
    <xsd:attribute default="생산이력" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:simpleType name="EnumCreationType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="생산"/>
        <xsd:enumeration value="접수"/>
        <xsd:enumeration value="NA"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="TypeRetentionPeriod">
    <xsd:sequence>
        <xsd:element name="RetentionPeriod">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="보존기간" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element minOccurs="0" name="ReasonofRetentionPeriod">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="보존기간책정사유" name="KDN" type="xsd:string"/>
```

```
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="보존기간정보" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="TypeRights">
    <xsd:sequence>
        <xsd:element minOccurs="0" name="SecurityCon">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="SecurityLevel">
                        <xsd:complexType>
                            <xsd:simpleContent>
                                <xsd:extension base="xsd:string">
                                    <xsd:attribute default="비밀분류" name="KDN" type="xsd:string"/>
                                </xsd:extension>
                            </xsd:simpleContent>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="MandateofSecurityLevel">
                        <xsd:complexType>
                            <xsd:simpleContent>
                                <xsd:extension base="xsd:string">
```

```
                        <xsd:attribute default="비밀분류근거" name="KDN" type="xsd:string"
                        />
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ProtectionPeriod">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="보호기간" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="비밀" name="KDN" type="xsd:string"/>
  </xsd:complexType>
</xsd:element>
<xsd:element minOccurs="0" name="InternalAccessScope">
  <xsd:complexType>
    <xsd:simpleContent>
      <xsd:extension base="EnumInternalAccessScopeType">
        <xsd:attribute default="접근범위" name="KDN" type="xsd:string"/>
      </xsd:extension>
```

```
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ExternalAccessCon">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="ExternalAccessControl">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="EnumExternalAccessControlType">
                                <xsd:attribute default="외부접근제어" name="KDN" type="xsd:string"
                                />
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element minOccurs="0" name="ExternalAccessReasonList">
                    <xsd:complexType>
                        <xsd:sequence>
                            <xsd:element maxOccurs="unbounded" minOccurs="0"
                                name="ExternalAccessReason">
                                <xsd:complexType>
                                    <xsd:simpleContent>
                                        <xsd:extension base="EnumExternalAccessReasonType">
                                          <xsd:attribute default="비공개사유" name="KDN"
```

```
                                type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
            <xsd:attribute default="비공개사유목록" name="KDN" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="0" name="LimitedContents">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="공개제한부분" name="KDN" type="xsd:string"
                    />
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="0" name="ExternalAccessDueDate">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:dateTime">
                    <xsd:attribute default="공개예정일시" name="KDN" type="xsd:string"
                    />
```

</xsd:extension> </xsd:simpleContent> </xsd:complexType> </xsd:element> <xsd:element minOccurs="0" name="MandateofExternalAccess"> <xsd:complexType> <xsd:simpleContent> <xsd:extension base="xsd:string"> <xsd:attribute default="공개관련근거" name="KDN" type="xsd:string" /> </xsd:extension> </xsd:simpleContent> </xsd:complexType> </xsd:element> <xsd:element maxOccurs="1" minOccurs="0" name="ListExternalAccessControl"> <xsd:annotation> <xsd:documentation>목록공개구분</xsd:documentation> </xsd:annotation> <xsd:complexType> <xsd:simpleContent><xsd:extension base="EnumListExternalAccessControlType"> <xsd:attribute fixed="목록공개구분" name="KDN" type="xsd:string"/> </xsd:extension> </xsd:simpleContent> </xsd:complexType> </xsd:element>	⑰ 목록의 공개 여부를 구분함.

<xsd:element minOccurs="0" name="ListExternalAccessReason" maxOccurs="unbounded"> <xsd:annotation> <xsd:documentation>목록비공개근거</xsd:documentation> </xsd:annotation> <xsd:complexType> <xsd:simpleContent> <xsd:extension base="EnumListExternalAccessReasonType"> <xsd:attribute fixed="목록비공개근거" name="KDN" type="xsd:string" /> </xsd:extension> </xsd:simpleContent> </xsd:complexType> </xsd:element> <xsd:element maxOccurs="1" minOccurs="0" name="ListNonDisclosureReason"> <xsd:annotation> <xsd:documentation>목록공개제한사유</xsd:documentation> </xsd:annotation> <xsd:complexType> <xsd:simpleContent> <xsd:extension base="xsd:string"> <xsd:attribute fixed="목록공개제한사유" name="KDN" type="xsd:string"/> </xsd:extension> </xsd:simpleContent>	⑱ 비공개일 경우 그 근거를 기록함. ⑲ 비공개일 경우, 구체적인 사유를 기술함.

```
                        </xsd:complexType>
                    </xsd:element>
                </xsd:sequence>
                <xsd:attribute default="공개" name="KDN" type="xsd:string"/>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="권한" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:simpleType name="EnumInternalAccessScopeType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="전체열람"/>
        <xsd:enumeration value="목록열람"/>
        <xsd:enumeration value="열람불가"/>
        <xsd:enumeration value="전행정기관 열람"/>
        <xsd:enumeration value="부서내공유"/>
        <xsd:enumeration value="담당자공유(비공유)"/>
        <xsd:enumeration value="기타"/>
        <xsd:enumeration value="NA"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="EnumExternalAccessControlType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="공개"/>
        <xsd:enumeration value="비공개"/>
```

```
            <xsd:enumeration value="30년"/>
            <xsd:enumeration value="15년"/>
            <xsd:enumeration value="10년"/>
            <xsd:enumeration value="5년"/>
            <xsd:enumeration value="3년"/>
            <xsd:enumeration value="기간설정"/>
            <xsd:enumeration value="일정공개"/>
            <xsd:enumeration value="일정공개전속"/>
            <xsd:enumeration value="부분공개"/>
            <xsd:enumeration value="비공개전속"/>
            <xsd:enumeration value="비공식"/>
            <xsd:enumeration value="1그룹일정"/>
            <xsd:enumeration value="2그룹일정"/>
            <xsd:enumeration value="3그룹일정"/>
            <xsd:enumeration value="내부보고"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumExternalAccessReasonType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="1호"/>
            <xsd:enumeration value="2호"/>
            <xsd:enumeration value="3호"/>
            <xsd:enumeration value="4호"/>
            <xsd:enumeration value="5호"/>
```

```
            <xsd:enumeration value="6호"/>
            <xsd:enumeration value="7호"/>
            <xsd:enumeration value="8호"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumEventType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="생산기관간 인계"/>
            <xsd:enumeration value="생산기관간 인수"/>
            <xsd:enumeration value="기록관으로 이관"/>
            <xsd:enumeration value="기록관에서 인수"/>
            <xsd:enumeration value="영구기록물관리기관으로 이관"/>
            <xsd:enumeration value="영구기록물관리기관에서 인수"/>
            <xsd:enumeration value="공개재분류"/>
            <xsd:enumeration value="공개 재분류"/>
            <xsd:enumeration value="보존기간 재책정"/>
            <xsd:enumeration value="보류"/>
            <xsd:enumeration value="폐기"/>
            <xsd:enumeration value="비밀 재분류"/>
            <xsd:enumeration value="접근범위 재책정"/>
            <xsd:enumeration value="기록물 재편철"/>
            <xsd:enumeration value="메타데이터 수정"/>
            <xsd:enumeration value="목록공개 변경"/>
            <xsd:enumeration value="기타"/>
```

```
            <xsd:enumeration value="기록관간 이관"/>
            <xsd:enumeration value="폐기결정"/>
            <xsd:enumeration value="폐기실행"/>
            <xsd:enumeration value="대체사본 보존결정"/>
            <xsd:enumeration value="기록물관리기관에서 추가등록"/>
            <xsd:enumeration value="기술입력"/>
            <xsd:enumeration value="폐기금지 변경"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:complexType name="TypeManagementHistory">
        <xsd:sequence>
            <xsd:element name="EventType">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="EnumEventType">
                            <xsd:attribute default="관리유형" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="EventDescription" minOccurs="0">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
```

```
                        <xsd:attribute default="관리설명" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="EventDateTime">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:dateTime">
                        <xsd:attribute default="관리일시" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="EventAgentIDRef">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:IDREF">
                        <xsd:attribute default="관리행위자" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="관리이력" name="KDN" type="xsd:string"/>
```

```
</xsd:complexType>
<xsd:complexType name="TypeUseHistory">
    <xsd:sequence>
        <xsd:element name="UseType">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="EnumUseType">
                        <xsd:attribute default="이용유형" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="UseDescription">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="이용설명" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="UseDateTime">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:dateTime">
```

```
                        <xsd:attribute default="이용일시" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="UseAgentIDRef">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:IDREF">
                        <xsd:attribute default="이용자" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="이용이력" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="TypePreservationHistory">
    <xsd:sequence>
        <xsd:element name="PreservationActionType">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="EnumPreservationActionType">
                        <xsd:attribute default="보존처리유형" name="KDN" type="xsd:string"/>
                    </xsd:extension>
```

```
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="PreservationActionDescription" minOccurs="0">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="보존처리설명" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="PreservationActionDate">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:dateTime">
                    <xsd:attribute default="보존처리일시" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="PreservationActionAgentIDRef">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:IDREF">
```

```
                        <xsd:attribute default="보존처리자" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="보존이력" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="TypeRelation">
    <xsd:sequence>
        <xsd:element name="RelationType" minOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="EnumRelationType">
                        <xsd:attribute default="관계유형" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="RelationItemID " maxOccurs="unbounded">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="관계대상식별자" name="KDN" type="xsd:string"/>
                    </xsd:extension>
```

```
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="RelationDescription" minOccurs="0">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="관계설명" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute default="관계정보" name="KDN" type="xsd:string"/>
    </xsd:complexType
    <xsd:simpleType name="EnumUseType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="전체열람"/>
            <xsd:enumeration value="목록열람"/>
            <xsd:enumeration value="다운로드"/>
            <xsd:enumeration value="복사"/>
            <xsd:enumeration value="허가받지 않은 접근"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
```

```
<xsd:simpleType name="EnumRelationTargetType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="지시사항"/>
        <xsd:enumeration value="문서보고"/>
        <xsd:enumeration value="메모보고"/>
        <xsd:enumeration value="행정부처보고"/>
        <xsd:enumeration value="국정과제"/>
        <xsd:enumeration value="자료유통"/>
        <xsd:enumeration value="기타"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="EnumSystemIDType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="전자기록생산시스템"/>
        <xsd:enumeration value="기록관리시스템"/>
        <xsd:enumeration value="영구기록관리시스템"/>
        <xsd:enumeration value="NA"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="EnumSecondaryIDType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="UCI"/>
        <xsd:enumeration value="URL"/>
        <xsd:enumeration value="URI"/>
        <xsd:enumeration value="URN"/>
```

```
            <xsd:enumeration value="DOI"/>
            <xsd:enumeration value="ISBN"/>
            <xsd:enumeration value="ISSN"/>
            <xsd:enumeration value="UUID"/>
            <xsd:enumeration value="생산기관 생산등록번호"/>
            <xsd:enumeration value="문서과 배부번호"/>
            <xsd:enumeration value="구기록물 문서번호"/>
            <xsd:enumeration value="지시사항 관리번호"/>
            <xsd:enumeration value="전자심의 관리번호"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:complexType name="TypeResult">
        <xsd:sequence>
            <xsd:element name="ResultType">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="EnumResultType">
                            <xsd:attribute default="실적유형" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="ResultTitle">
                <xsd:complexType>
```

```
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="실적제목" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ResultDateTime">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:dateTime">
                    <xsd:attribute default="실적일" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ResultAgentIDRef">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:IDREF">
                    <xsd:attribute default="실적생성자ID" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
```

```
        </xsd:sequence>
        <xsd:attribute default="실적정보" name="KDN" type="xsd:string"/>
    </xsd:complexType>
    <xsd:simpleType name="EnumResultType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="일지"/>
            <xsd:enumeration value="문서보고"/>
            <xsd:enumeration value="메모보고"/>
            <xsd:enumeration value="지시사항"/>
            <xsd:enumeration value="기타"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:complexType name="TypeWorkHandBooks">
        <xsd:sequence>
            <xsd:element name="HandBookType">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="EnumHandBookType">
                            <xsd:attribute default="업무편람유형" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="HandBookFilename">
```

```
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="업무편람대상파일" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute default="업무편람정보" name="KDN" type="xsd:string"/>
    </xsd:complexType>
    <xsd:simpleType name="EnumHandBookType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="관련법규"/>
            <xsd:enumeration value="예산"/>
            <xsd:enumeration value="업무메뉴얼"/>
            <xsd:enumeration value="관련메뉴얼"/>
            <xsd:enumeration value="자산정보자료"/>
            <xsd:enumeration value="업무참고자료"/>
            <xsd:enumeration value="해외자료"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumComponentType">
        <xsd:restriction base="xsd:string">
```

```
<xsd:enumeration value="관련법규"/>
<xsd:enumeration value="예산"/>
<xsd:enumeration value="업무메뉴얼"/>
<xsd:enumeration value="관련메뉴얼"/>
<xsd:enumeration value="자산정보자료"/>
<xsd:enumeration value="업무참고자료"/>
<xsd:enumeration value="해외자료"/>
<xsd:enumeration value="문서본문"/>
<xsd:enumeration value="문서첨부"/>
<xsd:enumeration value="보고경로첨부"/>
<xsd:enumeration value="메모첨부"/>
<xsd:enumeration value="수신자의견"/>
<xsd:enumeration value="지시첨부"/>
<xsd:enumeration value="회의체첨부"/>
<xsd:enumeration value="회의록첨부"/>
<xsd:enumeration value="회의결과첨부"/>
<xsd:enumeration value="안건결과첨부"/>
<xsd:enumeration value="일지첨부"/>
<xsd:enumeration value="대통령일정첨부"/>
<xsd:enumeration value="발언록"/>
<xsd:enumeration value="사후평가첨부"/>
<xsd:enumeration value="업무일지첨부"/>
<xsd:enumeration value="문서요지붙임"/>
<xsd:enumeration value="붙임첨부"/>
<xsd:enumeration value="본문"/>
```

```
            <xsd:enumeration value="첨부"/>
            <xsd:enumeration value="의견첨부"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumComponentPreservationType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="보존포맷변환"/>
            <xsd:enumeration value="바이러스검출"/>
            <xsd:enumeration value="바이러스치료"/>
            <xsd:enumeration value="특이사항"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumAgentType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="기안자"/>
            <xsd:enumeration value="검토자"/>
            <xsd:enumeration value="협조자"/>
            <xsd:enumeration value="결재자"/>
            <xsd:enumeration value="지시자"/>
            <xsd:enumeration value="수신자"/>
            <xsd:enumeration value="시행수신자"/>
            <xsd:enumeration value="발신자"/>
            <xsd:enumeration value="인계자"/>
```

<xsd:enumeration value="인수자"/> <xsd:enumeration value="보존행위자"/> <xsd:enumeration value="보고수신자"/> <xsd:enumeration value="수정자"/> <xsd:enumeration value="접수자"/> <xsd:enumeration value="평가자"/> <xsd:enumeration value="심사자"/> <xsd:enumeration value="이관승인자"/> <xsd:enumeration value="사용자"/> <xsd:enumeration value="서명자"/> <xsd:enumeration value="과제담당자"/> <xsd:enumeration value="과제관계자"/> <xsd:enumeration value="내부관계자"/> <xsd:enumeration value="과제담당자(이력)"/> <xsd:enumeration value="작성자"/> <xsd:enumeration value="기록물관리자"/> <xsd:enumeration value="주관자"/> <xsd:enumeration value="이행자"/> <xsd:enumeration value="참조자"/> <xsd:enumeration value="참석자"/> <xsd:enumeration value="업무담당자"/> <xsd:enumeration value="본문작성자"/> <xsd:enumeration value="요청자"/> <xsd:enumeration value="생산기관"/> <xsd:enumeration value="업무관리자"/>	

<xsd:enumeration value="내부참석자"/> <xsd:enumeration value="내부배석자"/> <xsd:enumeration value="내부열람자"/> <xsd:enumeration value="외부참석자"/> <xsd:enumeration value="외부배석자"/> <xsd:enumeration value="외부열람자"/> <xsd:enumeration value="주재자"/> <xsd:enumeration value="관계자"/> <xsd:enumeration value="처리자"/> <xsd:enumeration value="보고자"/> <xsd:enumeration value="의견작성자"/> <xsd:enumeration value="지시담당자"/> <xsd:enumeration value="담당자"/> <xsd:enumeration value="공유자"/> <xsd:enumeration value="기타"/> <xsd:enumeration value="병렬협조자"/> <xsd:enumeration value="병렬검토자"/> <xsd:enumeration value="1인결재자"/> <xsd:enumeration value="1인전결자"/> <xsd:enumeration value="1인대결자"/> <xsd:enumeration value="전결자"/> <xsd:enumeration value="대결자"/> <xsd:enumeration value="전대결자"/> <xsd:enumeration value="열람자"/> <xsd:enumeration value="생산부서"/>	

```
            <xsd:enumeration value="폐기금지/해제 이행자"/>
            <xsd:enumeration value="폐기금지/해제 발령자"/>
            <xsd:enumeration value="이용자"/>
            <xsd:enumeration value="보고행위자"/>
            <xsd:enumeration value="관리행위자"/>
            <xsd:enumeration value="보존행위자"/>
            <xsd:enumeration value="실적생성자"/>
            <xsd:enumeration value="부기정정요청자"/>
            <xsd:enumeration value="부기정정접수자"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:complexType name="TypeCorporateInfo">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="CorporateName">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="기관명" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element minOccurs="0" name="CorporateCode" nillable="false">
                <xsd:complexType>
```

```
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="기관코드" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
    </xsd:complexType>
    <xsd:complexType name="TypeSectionInfo">
        <xsd:sequence>
            <xsd:element minOccurs="0" name="SectionName">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="부서명" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element minOccurs="0" name="SectionCode">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="부서코드" name="KDN" type="xsd:string"/>
```

```
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>
<xsd:complexType name="TypePersonalInfo">
    <xsd:sequence>
        <xsd:element minOccurs="0" name="PersonName">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="개인명" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element minOccurs="0" name="PersonCode">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="UserCodeType">
                        <xsd:attribute default="개인코드" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
```

```
</xsd:element>
<xsd:element minOccurs="0" name="PersonID">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="UserIDType">
                <xsd:attribute default="개인ID" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
<xsd:element minOccurs="0" name="PositionName">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute default="직위(직급)명" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
<xsd:element minOccurs="0" name="PositionCode">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute default="직위(직급)코드" name="KDN" type="xsd:string"/>
            </xsd:extension>
```

```
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element minOccurs="0" name="PositionRole">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="역할구분" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element minOccurs="0" name="PositionEtc">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="추가정보" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
</xsd:complexType>
<xsd:simpleType name="EnumAlternativeTitleType">
    <xsd:restriction base="xsd:string">
```

```
            <xsd:enumeration value="부제목"/>
            <xsd:enumeration value="총서명"/>
            <xsd:enumeration value="부가제목"/>
            <xsd:enumeration value="정정제목"/>
            <xsd:enumeration value="영문제목"/>
            <xsd:enumeration value="기타"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumDescriptionType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="처리과 내용요약"/>
            <xsd:enumeration value="기록관 내용요약"/>
            <xsd:enumeration value="영구기록물관리기관 기술"/>
            <xsd:enumeration value="내용요약"/>
            <xsd:enumeration value="기술정보"/>
            <xsd:enumeration value="관리정보"/>
            <xsd:enumeration value="추진경과"/>
            <xsd:enumeration value="목적"/>
            <xsd:enumeration value="행사내용"/>
            <xsd:enumeration value="문서요지"/>
            <xsd:enumeration value="메모보고 본문"/>
            <xsd:enumeration value="기타"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
```

```
</xsd:simpleType>
<xsd:simpleType name="EnumPreservationActionType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="매체수록(광디스크)"/>
        <xsd:enumeration value="매체수록(M/F)"/>
        <xsd:enumeration value="매체수록(아카이빙 스토리지)"/>
        <xsd:enumeration value="소독"/>
        <xsd:enumeration value="제본"/>
        <xsd:enumeration value="탈산"/>
        <xsd:enumeration value="복원"/>
        <xsd:enumeration value="기타"/>
        <xsd:enumeration value="바이러스검출"/>
        <xsd:enumeration value="바이러스치료"/>
        <xsd:enumeration value="훼손사항"/>
        <xsd:enumeration value="보존포맷변환"/>
        <xsd:enumeration value="보존포맷 변환"/>
        <xsd:enumeration value="문서보존포맷변환"/>
        <xsd:enumeration value="장기보존포맷변환"/>
        <xsd:enumeration value="문서보존포맷 변환"/>
        <xsd:enumeration value="장기보존포맷 변환"/>
        <xsd:enumeration value="NA"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="UserCodeType">
    <xsd:restriction base="xsd:string">
```

```
            <xsd:maxLength value="40"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="UserIDType">
        <xsd:restriction base="xsd:string">
            <xsd:maxLength value="50"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="SectionCodeType">
        <xsd:restriction base="xsd:string"> </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="CorporateCodeType">
        <xsd:restriction base="xsd:string">
            <xsd:length value="7"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:complexType name="TypeRecordStorage">
        <xsd:sequence>
            <xsd:element name="TypeCon">
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element maxOccurs="unbounded" name="RecordType">
                            <xsd:complexType>
                                <xsd:simpleContent>
                                    <xsd:extension base="EnumRecordType">
```

```
                    <xsd:attribute default="기록유형" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="0" name="TypeOfCopy">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="EnumTypeOfCopy">
                    <xsd:attribute default="사본유형" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="0" name="AlternativeType">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="EnumAlternativeType">
                    <xsd:attribute default="기타유형" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="0" name="ComponentType">
        <xsd:complexType>
```

```
                        <xsd:simpleContent>
                            <xsd:extension base="EnumComponentType">
                                <xsd:attribute fixed="컴포넌트 유형" name="KDN" type="xsd:string"
                                />
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
            <xsd:attribute default="유형정보" name="KDN" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="0" name="FormatCon" maxOccurs="unbounded">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="FormatName">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="포맷명" name="KDN" type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="FormatVersion">
```

```
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="포맷버전" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="CreatingApplicationName">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="생성어플리케이션명" name="KDN"
                        type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="CreatingApplicationVersion">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="생성어플리케이션버전" name="KDN"
                        type="xsd:string"/>
                </xsd:extension>
```

```
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
            <xsd:attribute default="포맷" name="KDN" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="0" name="Medium" maxOccurs="unbounded">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="EnumMediumType">
                    <xsd:attribute default="저장매체" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element maxOccurs="unbounded" name="ExtentCon">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="ExtentType">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="EnumExtentType">
                                <xsd:attribute default="크기유형" name="KDN" type="xsd:string"/>
                            </xsd:extension>
```

```
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="Size">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="용량" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="Unit">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="EnumUnitType">
                            <xsd:attribute default="단위" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute default="크기" name="KDN" type="xsd:string"/>
    </xsd:complexType>
</xsd:element>
```

```
<xsd:element maxOccurs="1" minOccurs="0" name="LocationCon">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="Location">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="소장처" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="StorageDetails">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="소장위치" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute default="위치" name="KDN" type="xsd:string"/>
    </xsd:complexType>
</xsd:element>
```

```
<xsd:element maxOccurs="unbounded" minOccurs="0" name="StructureCon">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="StructureType">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="EnumStructureType">
                            <xsd:attribute default="유형" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element maxOccurs="unbounded" name="StructureInfoCon">
                <xsd:complexType>
                    <xsd:sequence>
                        <xsd:element name="TargetObjectID">
                            <xsd:complexType>
                                <xsd:simpleContent>
                                    <xsd:extension base="xsd:string">
                                      <xsd:attribute default="대상식별자" name="KDN"
                                      type="xsd:string"/>
                                    </xsd:extension>
                                </xsd:simpleContent>
                            </xsd:complexType>
                        </xsd:element>
```

```
                                <xsd:element name="TargetObjectInfo">
                                    <xsd:complexType>
                                        <xsd:simpleContent>
                                            <xsd:extension base="xsd:string">
                                              <xsd:attribute default="기록물 제목" name="KDN"
                                              type="xsd:string"/>
                                            </xsd:extension>
                                        </xsd:simpleContent>
                                    </xsd:complexType>
                                </xsd:element>
                            </xsd:sequence>
                            <xsd:attribute default="구조정보" name="KDN" type="xsd:string"/>
                        </xsd:complexType>
                    </xsd:element>
                </xsd:sequence>
                <xsd:attribute default="구조" name="KDN" type="xsd:string"/>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="저장정보" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="TypeClassification">
    <xsd:sequence>
        <xsd:element name="ClassificationType">
            <xsd:complexType>
```

```
            <xsd:simpleContent>
                <xsd:extension base="EnumClassificationType">
                    <xsd:attribute default="분류체계유형" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ClassificationValueCon" maxOccurs="unbounded" minOccurs="1">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="ClassificationID">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="분류ID" name="KDN" type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="ClassificationName">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="분류명" name="KDN" type="xsd:string"/>
                            </xsd:extension>
```

```
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="ClassificationLevel">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="EnumClassificationLevelType">
                                <xsd:attribute default="분류계층" name="KDN" type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
            <xsd:attribute default="분류값" name="KDN" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>
</xsd:sequence>
<xsd:attribute default="분류정보" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:simpleType name="EnumClassificationType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="BRM"/>
        <xsd:enumeration value="기록물분류기준"/>
        <xsd:enumeration value="공문서분류체계"/>
        <xsd:enumeration value="문서분류체계"/>
```

```
            <xsd:enumeration value="기타"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumRecordType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="일반문서"/>
            <xsd:enumeration value="간행물/도서류"/>
            <xsd:enumeration value="카드류"/>
            <xsd:enumeration value="도면/지도류"/>
            <xsd:enumeration value="사진필름류"/>
            <xsd:enumeration value="녹음동영상류"/>
            <xsd:enumeration value="행정박물"/>
            <xsd:enumeration value="웹기록물"/>
            <xsd:enumeration value="데이터세트"/>
            <xsd:enumeration value="전자우편"/>
            <xsd:enumeration value="기타"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumPreservationPlaceType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="기록관"/>
            <xsd:enumeration value="영구기록물관리기관"/>
            <xsd:enumeration value="NA"/>
```

```
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumReportRequestType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="기안"/>
            <xsd:enumeration value="검토"/>
            <xsd:enumeration value="협조"/>
            <xsd:enumeration value="병렬협조"/>
            <xsd:enumeration value="결재"/>
            <xsd:enumeration value="1인결재"/>
            <xsd:enumeration value="1인전결"/>
            <xsd:enumeration value="1인대결"/>
            <xsd:enumeration value="전결"/>
            <xsd:enumeration value="대결"/>
            <xsd:enumeration value="전대결"/>
            <xsd:enumeration value="재검토"/>
            <xsd:enumeration value="중단"/>
            <xsd:enumeration value="업무담당"/>
            <xsd:enumeration value="공석"/>
            <xsd:enumeration value="(공석)"/>
            <xsd:enumeration value="업무관리"/>
            <xsd:enumeration value="보고"/>
            <xsd:enumeration value="의견"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
```

```
</xsd:simpleType>
<xsd:simpleType name="EnumReportResultType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="기안"/>
        <xsd:enumeration value="검토"/>
        <xsd:enumeration value="협조"/>
        <xsd:enumeration value="병렬협조"/>
        <xsd:enumeration value="결재"/>
        <xsd:enumeration value="1인결재"/>
        <xsd:enumeration value="1인전결"/>
        <xsd:enumeration value="1인대결"/>
        <xsd:enumeration value="전결"/>
        <xsd:enumeration value="대결"/>
        <xsd:enumeration value="전대결"/>
        <xsd:enumeration value="재검토"/>
        <xsd:enumeration value="중단"/>
        <xsd:enumeration value="업무담당"/>
        <xsd:enumeration value="공석"/>
        <xsd:enumeration value="(공석)"/>
        <xsd:enumeration value="업무관리"/>
        <xsd:enumeration value="보고"/>
        <xsd:enumeration value="의견"/>
        <xsd:enumeration value="NA"/>
    </xsd:restriction>
</xsd:simpleType>
```

```
<xsd:simpleType name="EnumFileCVType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="직무상"/>
        <xsd:enumeration value="지시사항"/>
        <xsd:enumeration value="업무이관"/>
        <xsd:enumeration value="기타"/>
        <xsd:enumeration value="NA"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="EnumFileRoleType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="주관"/>
        <xsd:enumeration value="점검,조정"/>
        <xsd:enumeration value="협조,지원"/>
        <xsd:enumeration value="NA"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="TypeRecordSub">
    <xsd:complexContent mixed="false">
        <xsd:extension base="xsd:anyType">
            <xsd:attribute default="기록유형별 메타데이터" name="KDN" type="xsd:string"/>
        </xsd:extension>
    </xsd:complexContent>
</xsd:complexType>
<xsd:simpleType name="EnumListExternalAccessControlType">
```

```
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="공개"/>
            <xsd:enumeration value="비공개"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumListExternalAccessReasonType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="1호"/>
            <xsd:enumeration value="2호"/>
            <xsd:enumeration value="3호"/>
            <xsd:enumeration value="4호"/>
            <xsd:enumeration value="5호"/>
            <xsd:enumeration value="6호"/>
            <xsd:enumeration value="7호"/>
            <xsd:enumeration value="8호"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:simpleType name="EnumDestructionFreezeType">
        <xsd:restriction base="xsd:string">
            <xsd:enumeration value="폐기금지"/>
            <xsd:enumeration value="폐기금지해제"/>
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
```

```
</xsd:simpleType>
<xsd:simpleType name="EnumComponentTypeCreated">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="본문"/>
        <xsd:enumeration value="첨부"/>
        <xsd:enumeration value="NA"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="EnumCorrectionType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="부기"/>
        <xsd:enumeration value="정정"/>
        <xsd:enumeration value="NA"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:simpleType name="EnumRecordCreationSystemType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="업무관리시스템"/>
        <xsd:enumeration value="전자문서시스템"/>
        <xsd:enumeration value="구전자문서시스템"/>
        <xsd:enumeration value="행정정보시스템"/>
        <xsd:enumeration value="기록관리시스템"/>
        <xsd:enumeration value="자료관시스템"/>
        <xsd:enumeration value="업무관리시스템(08년이후)"/>
        <xsd:enumeration value="기타"/>
```

```
            <xsd:enumeration value="NA"/>
        </xsd:restriction>
    </xsd:simpleType>
    <xsd:complexType name="TypeDestructionFreeze">
        <xsd:sequence>
            <xsd:element name="DestructionFreezeOrder">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute fixed="폐기금지 이력 순번" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="DestructionFreezeType">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="EnumDestructionFreezeType">
                            <xsd:attribute fixed="폐기금지 유형" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="DestructionFreezeDescription">
                <xsd:complexType>
```

```
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute fixed="폐기금지/해제 설명" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="DestructionFreezeIssueDate">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:dateTime">
                    <xsd:attribute fixed="폐기금지/해제 발령일자" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element minOccurs="0" name="DestructionFreezeIssuerIDRef">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:IDREF">
                    <xsd:attribute fixed="폐기금지/해제 발령자" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
```

```
            <xsd:element minOccurs="0" name="DestructionFreezeImplementDateTime">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:dateTime">
                            <xsd:attribute fixed="폐기금지/해제 이행일시" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element minOccurs="0" name="DestructionFreezePerformerIDRef">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:IDREF">
                            <xsd:attribute fixed="폐기금지/해제 이행자" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute fixed="기록물 폐기금지" name="KDN" type="xsd:string"/>
    </xsd:complexType>
    <xsd:complexType name="TypeIntegrityCheck">
        <xsd:sequence>
            <xsd:element name="IntegrityCheckName">
                <xsd:complexType>
```

```
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute fixed="무결성체크법" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="IntegrityCheckValue">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute fixed="무결성체크값" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute fixed="무결성체크" name="KDN" type="xsd:string"/>
    </xsd:complexType>
    <xsd:complexType name="TypeCreater">
        <xsd:sequence>
            <xsd:element maxOccurs="unbounded" name="CreatorIDRef">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:IDREF">
```

```
                        <xsd:attribute default="생산자ID참조" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="생산자목록" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:simpleType name="EnumRelationType">
    <xsd:restriction base="xsd:string">
        <xsd:enumeration value="포함한다"/>
        <xsd:enumeration value="포함된다"/>
        <xsd:enumeration value="선행"/>
        <xsd:enumeration value="후행"/>
        <xsd:enumeration value="사본"/>
        <xsd:enumeration value="원본"/>
        <xsd:enumeration value="포맷변환"/>
        <xsd:enumeration value="참조"/>
        <xsd:enumeration value="기타"/>
        <xsd:enumeration value="NA"/>
    </xsd:restriction>
</xsd:simpleType>
<xsd:complexType name="Type_PreservationDescInfoForDataset">
    <xsd:annotation>
        <xsd:documentation>[문서 유형 장기보존 메타데이터] 장기보존패키지 기록물건,기록물철 객체 스키마
```

```
            장기보존메타데이터</xsd:documentation>
        </xsd:annotation>
        <xsd:sequence>
            <xsd:element name="ObjectCon" type="Type_ObjectCon" minOccurs="1" maxOccurs="1"/>
            <xsd:element name="CreatorCon" type="Type_CreatorCon" minOccurs="1"
                maxOccurs="unbounded"/>
            <xsd:element name="RecordManagementID" minOccurs="1" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="기록관리식별자" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="RecordTitle" minOccurs="1" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="기록물명" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="SystemInfoCon" type="Type_SystemInfoCon" minOccurs="1" maxOccurs="1"/>
```

⑳ [문서형 장기보존메타데이터]는 장기보존패키지 내 기록물건·철 객체에 대한 식별 및 기술 정보를 정의.

주요 구성요소

- ObjectCon: 객체의 기본 속성 및 구조 정의
- CreatorCon: 기록물 생성 주체 정보
- RecordManagementID: 고유 기록관리 식별자

```
            <xsd:element name="DatasetInfoCon" type="Type_DatasetInfoCon" minOccurs="1"
                maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:attribute default="데이터세트 유형 장기보존 메타데이터" name="KDN" type="xsd:string"/>
    </xsd:complexType>
    <xsd:complexType name="Type_ObjectCon">
        <xsd:sequence>
            <xsd:element name="ObjectType" minOccurs="1" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="객체유형" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="ObjectVersion" minOccurs="1" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="객체버전" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
```

```
<xsd:element name="ObjectTypeDescription" minOccurs="1" maxOccurs="1">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute default="객체유형설명" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
<xsd:element name="ObjectCreationDate" minOccurs="1" maxOccurs="1">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute default="객체생성일시" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
<xsd:element name="PackagingScope" minOccurs="1" maxOccurs="1">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute default="패키징범위" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
```

```
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="객체메타데이터" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="Type_CreatorCon">
    <xsd:sequence>
        <xsd:element name="CreatorType" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="생산자유형" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="CorporateName" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="기관명" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
```

```
<xsd:element name="CorporateCode" minOccurs="0" maxOccurs="1">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute default="기관코드" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
<xsd:element name="SectionName" minOccurs="0" maxOccurs="1">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute default="부서명" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
<xsd:element name="SectionCode" minOccurs="0" maxOccurs="1">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute default="부서코드" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
```

```
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="PersonName" minOccurs="0" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="개인명" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="생산자" name="KDN" type="xsd:string"/>
</xsd:complexType>

<xsd:complexType name="Type_SystemInfoCon">
    <xsd:sequence>
        <xsd:element name="SystemName" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="시스템명" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
```

```
</xsd:element>
<xsd:element name="SystemID" minOccurs="1" maxOccurs="1">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute default="시스템식별자" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
<xsd:element name="SystemOverview" minOccurs="0" maxOccurs="1">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute default="시스템개요" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
<xsd:element name="SystemImplementationYear" minOccurs="1" maxOccurs="1">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute default="시스템구축연도" name="KDN" type="xsd:string"/>
            </xsd:extension>
```

```
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="SystemDecommissioningYear" minOccurs="0" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="시스템폐기연도" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="DBMSInfoCon" type="Type_DBMSInfoCon" minOccurs="1"
                maxOccurs="unbounded"/>
            <xsd:element name="InterfacedSystemCon" type="Type_InterfacedSystemCon" minOccurs="0"
                maxOccurs="unbounded"/>
            <xsd:element name="DeliverableCon" type="Type_DeliverableCon" minOccurs="0"
                maxOccurs="unbounded"/>
        </xsd:sequence>
        <xsd:attribute default="시스템정보" name="KDN" type="xsd:string"/>
    </xsd:complexType
    <xsd:complexType name="Type_DatasetInfoCon">
        <xsd:sequence>
            <xsd:element name="UnitFunctionCon" type="Type_UnitFunctionCon" minOccurs="1"
                maxOccurs="1"/>
```

```
        <xsd:element name="UnitFunctionScopeCon" type="Type_UnitFunctionScopeCon" minOccurs="1"
            maxOccurs="unbounded"/>
        <xsd:element name="RightsCon" type="Type_RightsCon" minOccurs="1" maxOccurs="unbounded"/>
        <xsd:element name="UsagePeriodCon" type="Type_UsagePeriodCon" minOccurs="1"
            maxOccurs="1"/>
        <xsd:element name="RetentionPeriodCon" type="Type_RetentionPeriodCon" minOccurs="1"
            maxOccurs="1"/>
        <xsd:element name="DispositionCon" type="Type_DispositionCon" minOccurs="0"
            maxOccurs="1"/>
        <xsd:element name="Medium" minOccurs="0" maxOccurs="unbounded">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="매체" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ManagementHistoryCon" type="Type_ManagementHistoryCon" minOccurs="0"
            maxOccurs="unbounded"/>
        <xsd:element name="PreservationHistoryCon" type="Type_PreservationHistoryCon"
            minOccurs="0" maxOccurs="unbounded"/>
        <xsd:element name="ComponentCon" type="Type_ComponentCon" minOccurs="0"
            maxOccurs="unbounded"/>
    </xsd:sequence>
```

```
    <xsd:attribute default="데이터세트정보" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="Type_DBMSInfoCon">
    <xsd:sequence>
        <xsd:element name="DBMSNameVersion" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="DBMS명/버전" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="FunctionDBName" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="업무DB명" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="DBMS정보" name="KDN" type="xsd:string"/>
</xsd:complexType>
```

```
<xsd:complexType name="Type_InterfacedSystemCon">
    <xsd:sequence>
        <xsd:element name="InterfacedType" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="연계유형" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="InterfacedSytemName" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="연계시스템명" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="InterfacedSytemID" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="연계시스템식별자" name="KDN" type="xsd:string"/>
```

```
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="InterfacedSystemCoporateName" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="연계시스템보유기관명" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="InterfacedSystemCoporateID" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="연계시스템보유기관코드" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="InterfaceFormCon" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
            <xsd:sequence>
```

```
                    <xsd:element name="InterfaceMethod" minOccurs="1" maxOccurs="1">
                        <xsd:complexType>
                            <xsd:simpleContent>
                                <xsd:extension base="xsd:string">
                                    <xsd:attribute default="연계방식" name="KDN" type="xsd:string"/>
                                </xsd:extension>
                            </xsd:simpleContent>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="InterfaceDirection" minOccurs="1" maxOccurs="1">
                        <xsd:complexType>
                            <xsd:simpleContent>
                                <xsd:extension base="xsd:string">
                                    <xsd:attribute default="연계방향" name="KDN" type="xsd:string"/>
                                </xsd:extension>
                            </xsd:simpleContent>
                        </xsd:complexType>
                    </xsd:element>
                </xsd:sequence>
                <xsd:attribute default="연계형태" name="KDN" type="xsd:string"/>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="연계시스템" name="KDN" type="xsd:string"/>
</xsd:complexType>
```

```
<xsd:complexType name="Type_DeliverableCon">
    <xsd:sequence>
        <xsd:element name="DeliverableType" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="산출물유형" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="DeliverableTitle" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="산출물제목" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="DeliverableFormatCon" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="FormatName" minOccurs="1" maxOccurs="1">
                        <xsd:complexType>
```

```
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="포맷명" name="KDN" type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="FormatVersion" minOccurs="0" maxOccurs="1">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="포맷버전 " name="KDN" type="xsd:string"
                                />
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
            <xsd:attribute default="산출물포맷" name="KDN" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="DeliverableSize" minOccurs="1" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
```

```
                    <xsd:attribute default="산출물크기" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="DeliverableLocation" minOccurs="1" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="산출물위치" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ManifestIDRef" minOccurs="1" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="ManifestID참조" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="DeliverableCIID" minOccurs="1" maxOccurs="1">
        <xsd:complexType>
```

```
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="산출물CI식별자" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute default="산출물" name="KDN" type="xsd:string"/>
    </xsd:complexType>
    <xsd:complexType name="Type_UnitFunctionCon">
        <xsd:sequence>
            <xsd:element name="UnitFunctionID" minOccurs="0" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="단위기능식별자" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="UnitFunctionName" minOccurs="1" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
```

```
                        <xsd:attribute default="단위기능명" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="UnitFunctionOverview" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="단위기능개요" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="SubjectWords" minOccurs="0" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="주제어" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="단위기능정보" name="KDN" type="xsd:string"/>
```

```
</xsd:complexType>
<xsd:complexType name="Type_UnitFunctionScopeCon">
    <xsd:sequence>
        <xsd:element name="RetrievalQuery" minOccurs="0" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="DB질의문" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ApplicationScope" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="적용범위" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ApplicationScopeInfo" minOccurs="0" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
```

```
                        <xsd:attribute default="ApplicationScopeInfo" name="KDN"
                            type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="단위기능범위" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="Type_RightsCon">
    <xsd:sequence>
        <xsd:element name="DataOwingCorporateCon" minOccurs="1" maxOccurs="unbounded">
            <xsd:complexType>
                <xsd:sequence>
                    <xsd:element name="CorporateName" minOccurs="1" maxOccurs="1">
                        <xsd:complexType>
                            <xsd:simpleContent>
                                <xsd:extension base="xsd:string">
                                    <xsd:attribute default="기관명" name="KDN" type="xsd:string"/>
                                </xsd:extension>
                            </xsd:simpleContent>
                        </xsd:complexType>
                    </xsd:element>
                    <xsd:element name="CorporateCode" minOccurs="0" maxOccurs="1">
                        <xsd:complexType>
```

```
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="기관코드 " name="KDN" type="xsd:string"
                    />
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="SectionName" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="부서명 " name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="SectionCode" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="부서코드 " name="KDN" type="xsd:string"
                    />
                </xsd:extension>
            </xsd:simpleContent>
```

```
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
            <xsd:attribute default="데이터보유권한" name="KDN" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="SystemManagementCorporateCon" minOccurs="1" maxOccurs="unbounded">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="CorporateName" minOccurs="1" maxOccurs="1">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="기관명" name="KDN" type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="CorporateCode" minOccurs="0" maxOccurs="1">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="기관코드 " name="KDN" type="xsd:string"
                                />
                            </xsd:extension>
```

```
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="SectionName" minOccurs="0" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="부서명 " name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="SectionCode" minOccurs="0" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="부서코드 " name="KDN" type="xsd:string"
                            />
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute default="시스템관리권한" name="KDN" type="xsd:string"/>
    </xsd:complexType>
```

```
</xsd:element>
<xsd:element name="InternalAccessScope" minOccurs="1" maxOccurs="1">
    <xsd:complexType>
        <xsd:simpleContent>
            <xsd:extension base="xsd:string">
                <xsd:attribute default="접근권한" name="KDN" type="xsd:string"/>
            </xsd:extension>
        </xsd:simpleContent>
    </xsd:complexType>
</xsd:element>
<xsd:element name="SecurityCon" minOccurs="0" maxOccurs="1">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="SecurityLevel" minOccurs="1" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="비밀분류" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="MandateofSecurityLevel" minOccurs="1" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
```

```
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="비밀분류근거 " name="KDN"
                                type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="ProtectionPeriod" minOccurs="1" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="보호기간" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute default="비밀" name="KDN" type="xsd:string"/>
    </xsd:complexType>
</xsd:element>
<xsd:element name="ExternalAccessCon" minOccurs="1" maxOccurs="unbounded">
    <xsd:complexType>
        <xsd:sequence>
            <xsd:element name="DisclosureType" minOccurs="1" maxOccurs="1">
                <xsd:complexType>
```

```
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="공개구분" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="NonDisclosureReason" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="비공개사유 " name="KDN" type="xsd:string"
                    />
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="LimitedContents" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="공개제한부분 " name="KDN"
                        type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
```

```
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="DisclosureLevel" minOccurs="0" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="공개대상계층" name="KDN" type="xsd:string"
                            />
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="DisclosureCoverage" minOccurs="0" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="공개대상범위" name="KDN" type="xsd:string"
                            />
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute default="공개" name="KDN" type="xsd:string"/>
    </xsd:complexType>
```

```
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute default="권한" name="KDN" type="xsd:string"/>
    </xsd:complexType>
    <xsd:complexType name="Type_UsagePeriodCon">
        <xsd:sequence>
            <xsd:element name="UsageStartDateTime" minOccurs="1" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="생산일시" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="UsageEndDateTime" minOccurs="0" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="종료일시" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
        </xsd:sequence>
```

```
    <xsd:attribute default="생산기간" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="Type_RetentionPeriodCon">
    <xsd:sequence>
        <xsd:element name="RetentionPeriod" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="보존기간" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ReasionOfRetentionPeriod" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="보존기간책정사유" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="RetentionStartDate" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
```

```
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="기산일" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="보존기간정보" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="Type_DispositionCon">
    <xsd:sequence>
        <xsd:element name="DispositionConstraint" minOccurs="0" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="처분제약사항" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="DispositionMethod" minOccurs="0" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="처분방법" name="KDN" type="xsd:string"/>
```

```
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="처분" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="Type_ManagementHistoryCon">
    <xsd:sequence>
        <xsd:element name="EventType" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="관리유형" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="EventDescription" minOccurs="0" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="관리설명" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
```

```
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="EventDateTime" minOccurs="1" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="관리일시" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="EventAgentCon" minOccurs="1" maxOccurs="1">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="CorporateName" minOccurs="1" maxOccurs="1">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="기관명" name="KDN" type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="CorporateCode" minOccurs="0" maxOccurs="1">
                    <xsd:complexType>
```

```
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="기관코드" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="SectionName" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="부서명" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="SectionCode" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="부서코드 " name="KDN" type="xsd:string"
                    />
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
```

```
                </xsd:element>
                <xsd:element name="PersonName" minOccurs="1" maxOccurs="1">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="개인명 " name="KDN" type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="PositionTitle" minOccurs="0" maxOccurs="1">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="직위(직급)명" name="KDN"
                                    type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
            <xsd:attribute default="관리일시" name="KDN" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ChangedElementCon" minOccurs="0" maxOccurs="unbounded">
```

```
<xsd:complexType>
    <xsd:sequence>
        <xsd:element name="ChangedElementName" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="변경요소명 " name="KDN" type="xsd:string"
                        />
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ChangedElementValue" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="변경이전값" name="KDN" type="xsd:string"
                        />
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
    <xsd:attribute default="변경요소 " name="KDN" type="xsd:string"/>
</xsd:complexType>
```

```
            </xsd:element>
        </xsd:sequence>
        <xsd:attribute default="관리이력" name="KDN" type="xsd:string"/>
    </xsd:complexType>
    <xsd:complexType name="Type_PreservationHistoryCon">
        <xsd:sequence>
            <xsd:element name="PreservationActionType" minOccurs="1" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="보존처리유형" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="PreservationActionDescription" minOccurs="0" maxOccurs="1">
                <xsd:complexType>
                    <xsd:simpleContent>
                        <xsd:extension base="xsd:string">
                            <xsd:attribute default="보존처리설명" name="KDN" type="xsd:string"/>
                        </xsd:extension>
                    </xsd:simpleContent>
                </xsd:complexType>
            </xsd:element>
            <xsd:element name="PreservationActionDate" minOccurs="1" maxOccurs="1">
```

```
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="보존처리일시" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="PreservationActionAgentCon" minOccurs="1" maxOccurs="1">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="CorporateName" minOccurs="1" maxOccurs="1">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="기관명 " name="KDN" type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="CorporateCode" minOccurs="0" maxOccurs="1">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="기관코드 " name="KDN" type="xsd:string"
```

```
                    />
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="SectionName" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="부서명" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="SectionCode" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="부서코드" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="PersonName" minOccurs="1" maxOccurs="1">
        <xsd:complexType>
```

```
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="개인명 " name="KDN" type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="PositionTitle" minOccurs="0" maxOccurs="1">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="직위(직급)명 " name="KDN"
                                    type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
            </xsd:sequence>
            <xsd:attribute default="보존행위자" name="KDN" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>
</xsd:sequence>
<xsd:attribute default="보존이력" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="Type_ComponentCon">
```

```
<xsd:sequence>
    <xsd:element name="ComponentType" minOccurs="1" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="컴포넌트유형" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ComponentTitle" minOccurs="1" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="컴포넌트제목" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ComponentOverview" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="컴포넌트개요" name="KDN" type="xsd:string"/>
                </xsd:extension>
```

```
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ComponentFormatCon" minOccurs="1" maxOccurs="1">
        <xsd:complexType>
            <xsd:sequence>
                <xsd:element name="FormatName" minOccurs="1" maxOccurs="1">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="포맷명 " name="KDN" type="xsd:string"/>
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="FormatVersion" minOccurs="0" maxOccurs="1">
                    <xsd:complexType>
                        <xsd:simpleContent>
                            <xsd:extension base="xsd:string">
                                <xsd:attribute default="포맷버전 " name="KDN" type="xsd:string"
                                />
                            </xsd:extension>
                        </xsd:simpleContent>
                    </xsd:complexType>
                </xsd:element>
```

```
            </xsd:sequence>
            <xsd:attribute default="컴포넌트포맷" name="KDN" type="xsd:string"/>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ComponentSize" minOccurs="1" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="컴포넌트크기 " name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ComponentLocation" minOccurs="1" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
                <xsd:extension base="xsd:string">
                    <xsd:attribute default="컴포넌트위치" name="KDN" type="xsd:string"/>
                </xsd:extension>
            </xsd:simpleContent>
        </xsd:complexType>
    </xsd:element>
    <xsd:element name="ComponentDetailLocation" minOccurs="0" maxOccurs="1">
        <xsd:complexType>
            <xsd:simpleContent>
```

```
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="컴포넌트세부위치" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ComponentCIID" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="컴포넌트CI식별자" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ManifestIDRef" minOccurs="1" maxOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="ManifestItemID참조" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
    </xsd:sequence>
```

```
    <xsd:attribute default="컴포넌트" name="KDN" type="xsd:string"/>
</xsd:complexType>
<xsd:complexType name="TypeContentItem">
    <xsd:sequence>
        <xsd:element name="ContentTitle" minOccurs="1">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="콘텐츠제목" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ContentMIME" minOccurs="0">
            <xsd:complexType>
                <xsd:simpleContent>
                    <xsd:extension base="xsd:string">
                        <xsd:attribute default="콘텐츠MIME" name="KDN" type="xsd:string"/>
                    </xsd:extension>
                </xsd:simpleContent>
            </xsd:complexType>
        </xsd:element>
        <xsd:element name="ManifestIDRef">
            <xsd:annotation>
                <xsd:documentation>패키지 Manifest 아이템에 대한 ID 참조</xsd:documentation>
```

</xsd:annotation> <xsd:complexType> <xsd:simpleContent> <xsd:extension base="xsd:string"> <xsd:attribute default="Manifest참조ID" name="KDN" type="xsd:string"/> </xsd:extension> </xsd:simpleContent> </xsd:complexType> </xsd:element> <xsd:element name="ContentSize" minOccurs="0"> <xsd:complexType> <xsd:simpleContent> <xsd:extension base="xsd:string"> <xsd:attribute default="콘텐츠크기" name="KDN" type="xsd:string"/> </xsd:extension> </xsd:simpleContent> </xsd:complexType> </xsd:element> <xsd:element minOccurs="0" name="ContentVersion"> <xsd:complexType> <xsd:simpleContent> <xsd:extension base="xsd:string"> <xsd:attribute default="콘텐츠버전" name="KDN" type="xsd:string"/> </xsd:extension> </xsd:simpleContent>	㉑ 장기보존 패키지 Manifest에 정의된 파일 또는 객체의 고유 ID를 참조하여 대상 파일을 식별함.

</xsd:complexType> </xsd:element> <xsd:element minOccurs="0" name="ContentHash"> <xsd:complexType> <xsd:simpleContent> <xsd:extension base="xsd:string"> <xsd:attribute default="콘텐츠해쉬" name="KDN" type="xsd:string"/> <xsd:attribute default="SHA2-512" name="HashAlgorithm" type="EnumHashAlgorithmType"/> </xsd:extension> </xsd:simpleContent> </xsd:complexType> </xsd:element> </xsd:sequence> <xsd:attribute name="ContentID" type="xsd:ID"> <xsd:annotation> <xsd:documentation>콘텐츠 정보 개별 아이템에 대한 ID로 기록물 Component 항목에서 참조</xsd:documentation> </xsd:annotation> </xsd:attribute> <xsd:attribute name="OrgContentID" type="xsd:IDREF"/> <xsd:attribute name="KDN" type="xsd:string" default="콘텐츠항목"/> </xsd:complexType> <xsd:complexType name="TypeDC_TotalDatasetInfo"> <xsd:annotation>	㉒ ContentID는 콘텐츠 정보의 개별 아이템에 부여되는 고유 식별자임

<xsd:documentation> ◎ 각 xml 요소에 값을 기입할 때, 장기보존 메타데이터와 연계된 element가 존재한다면, [차상위요소부터 전체 요소명]을 기입하고 실제 값을 기입한다. (예시) (1) Title의 경우, [기록물명]인사관리기록 (2) Creator의 경우, [생산자-기관명(데이터보유기관)]A기관;B기관 (3) Subject의 경우, [시스템정보-시스템명]A시스템;[데이터세트-단위기능정보-단위기능명]B기능;C기능;D기능;E기능;F기능 ◎ 하나의 DC 유형 요소에 다른 종류의 장기보존 메타데이터 요소값을 입력해야 하는 경우(예, Format 요소)에는, 같은 유형의 DC 요소를 여러 개 기입한다. (예시) (1) Format의 경우, <dc:format>[시스템정보-DBMS정보-DBMS명/버전]Oracle19c Enterprise Edition;MySQL10.0</dc:format><dc:format>시스템정보-산출물-산출물포맷-포맷명]hwp;pdf;xlsx;dax</dc:format></xsd:documentation> </xsd:annotation> <xsd:sequence> <xsd:choice minOccurs="0" maxOccurs="unbounded"> <xsd:element ref="dc:title"> </xsd:element> <xsd:element ref="dc:creator"> </xsd:element> <xsd:element ref="dc:subject"> </xsd:element> <xsd:element ref="dc:description"> </xsd:element> <xsd:element ref="dc:publisher"> </xsd:element> <xsd:element ref="dc:contributor"> </xsd:element> <xsd:element ref="dc:date"> </xsd:element> <xsd:element ref="dc:type"> </xsd:element>	㉓ 기입원칙: 1) DC 요소에 값을 기록할 때, 장기보존 메타데이터에서의 연계 위치를 각 값 앞에 [차상위요소부터 전체 요소명] 형태로 표기한 뒤 실제 값을 기입. 2) 단일 DC 요소에 서로 다른 종류의 장기보존 메타데이터 값을 함께 넣어야 할 경우 동일 DC 요소를 여러 번 반복.

<xsd:element ref="dc:format"> </xsd:element>
<xsd:element ref="dc:identifier"> </xsd:element>
<xsd:element ref="dc:source"> </xsd:element>
<xsd:element ref="dc:language"> </xsd:element>
<xsd:element ref="dc:relation"> </xsd:element>
<xsd:element ref="dc:coverage"> </xsd:element>
<xsd:element ref="dc:rights"> </xsd:element>
</xsd:choice>
</xsd:sequence>
<xsd:attribute name="type" type="xsd:string" use="required" fixed="데이터세트_전체정보"/>
<xsd:attribute name="scheme" type="xsd:string" use="required" fixed="DC"/>
</xsd:complexType>
<xsd:complexType name="TypeDC_SingleDatasetInfo">
<xsd:annotation>
<xsd:documentation>
◎ 각 xml 요소에 값을 기입할 때, 장기보존 메타데이타와 연계된 element가 존재한다면, [차상위요소부터 전체 요소명]을 기입하고 실제 값을 기입한다.
(예시)
(1) Title의 경우, [기록물명]인사관리기록
(2) Creator의 경우, [생산자-기관명(데이터보유기관)]A기관;B기관
(3) Subject의 경우, [시스템정보-시스템명]A시스템;[데이터세트-단위기능정보-단위기능명]B기능;C기능;D기능;E기능;F기능
◎ 하나의 DC 유형 요소에 다른 종류의 장기보존 메타데이터 요소값을 입력해야 하는 경우(예, Format 요소)에는, 같은 유형의 DC 요소를 여러 개 기입한다.
(예시)

(1) Format의 경우, <dc:format>[시스템정보-DBMS정보-DBMS명/버전]Oracle19c Enterprise Edition;MySQL10.0</dc:format><dc:format>시스템정보-산출물-산출물포맷-포맷명]hwp;pdf;xlsx;dax</dc:format></xsd:documentation>
`</xsd:annotation>`
`<xsd:sequence>`
`<xsd:choice minOccurs="0" maxOccurs="unbounded">`
`<xsd:element ref="dc:title"> </xsd:element>`
`<xsd:element ref="dc:creator"> </xsd:element>`
`<xsd:element ref="dc:subject"> </xsd:element>`
`<xsd:element ref="dc:description"> </xsd:element>`
`<xsd:element ref="dc:publisher"> </xsd:element>`
`<xsd:element ref="dc:contributor"> </xsd:element>`
`<xsd:element ref="dc:date"> </xsd:element>`
`<xsd:element ref="dc:type"> </xsd:element>`
`<xsd:element ref="dc:format"> </xsd:element>`
`<xsd:element ref="dc:identifier"> </xsd:element>`
`<xsd:element ref="dc:source"> </xsd:element>`
`<xsd:element ref="dc:language">`
`<xsd:annotation>`
`<xsd:documentation>◎ 'KO'로 고정값 설정</xsd:documentation>`
`</xsd:annotation>`
`</xsd:element>`
`<xsd:element ref="dc:relation"> </xsd:element>`
`<xsd:element ref="dc:coverage"> </xsd:element>`
`<xsd:element ref="dc:rights"> </xsd:element>`
	㉔ 데이터세트의 '단위정보'를 기술하기 위한 DC 구조. 상위 수준의

</xsd:choice> </xsd:sequence> <xsd:attribute name="type" type="xsd:string" use="required" fixed="데이터세트_단위정보"/> <xsd:attribute name="scheme" type="xsd:string" use="required" fixed="DC"/> </xsd:complexType> <xsd:complexType name="TypeDC_Default"> <xsd:annotation> <xsd:documentation>자유롭게 설정할 수 있음</xsd:documentation> </xsd:annotation> <xsd:sequence> <xsd:choice minOccurs="0" maxOccurs="unbounded"> <xsd:element ref="dc:title"/> <xsd:element ref="dc:creator"/> <xsd:element ref="dc:subject"/> <xsd:element ref="dc:description"/> <xsd:element ref="dc:publisher"/> <xsd:element ref="dc:contributor"/> <xsd:element ref="dc:date"/> <xsd:element ref="dc:type"/> <xsd:element ref="dc:format"/> <xsd:element ref="dc:identifier"/> <xsd:element ref="dc:source"/> <xsd:element ref="dc:language"/> <xsd:element ref="dc:relation"/> <xsd:element ref="dc:coverage"/>	총괄 메타데이터(TypeDC_TotalDatasetInfo)와 구분하여, 특정 기능 또는 단위 데이터세트의 개별 정보를 표현.

<xsd:element ref="dc:rights"/> </xsd:choice> </xsd:sequence> <xsd:attribute name="scheme" type="xsd:string" use="required" fixed="DC"/> </xsd:complexType> <xsd:complexType name="TypeEAD_TotalDatasetInfo"> <xsd:sequence> <xsd:element ref="ead:ead"/> </xsd:sequence> <xsd:attribute name="type" type="xsd:string" use="required" fixed="데이터세트_전체정보"/> <xsd:attribute name="scheme" type="xsd:string" use="required" fixed="EAD"/> </xsd:complexType> <xsd:complexType name="TypeEAD_SingleDatasetInfo"> <xsd:sequence> <xsd:element ref="ead:ead"/> </xsd:sequence> <xsd:attribute name="type" type="xsd:string" use="required" fixed="데이터세트_단위정보"/> <xsd:attribute name="scheme" type="xsd:string" use="required" fixed="EAD"/> </xsd:complexType> <xsd:complexType name="TypeEAD_Default"> <xsd:annotation> <xsd:documentation>자유롭게 설정할 수 있음</xsd:documentation> </xsd:annotation> <xsd:sequence> <xsd:element ref="ead:ead"/>	㉕ 데이터세트 전체 수준을 기술하기 위한 EAD 기반 컨테이너.

```
        </xsd:sequence>
        <xsd:attribute name="scheme" type="xsd:string" use="required" fixed="EAD"/>
    </xsd:complexType>
</xsd:schema>
```

참고문헌

[1] CCSDS. 2012년 6월. *Reference Model for an Open Archiveal Information System(OAIS).*

[2] 호주. 2003년. *Standard for the Management of Electronic Records PROS 99/007 : The Victorian Electronic Records Strategy(VERS)*

[3] 호주. 2003년. *99/007 Specification 3: VERS Standard Electronic Record Format : The Victorian Electronic Records Starategy(VERS)* <http://www.prov.vic.gov.au/vers/standard/ver1/99-7-3toc.htm>

[4] DILCIS Board. 2019년 5월 31일. *E-ARK Archival Information Package(AIP).*

[5] DILCIS Board. 2019년 10월 28일. *E-ARK DIP : Specification for Dissemination Information Packages.*

[6] DILCIS Board. 2020년 8월 1일. *E-ARK CSIP : Common Specification for Information Packages.*

[7] DILCIS Board. 2020년 8월 1일. *E-ARK SIP : Specification for Information Packages.*

[8] 한국 DOI센터. <https://www.doi.or.kr/wordpress/>

오픈 포맷을 활용한 행정정보 데이터세트
보존 방안 연구

[첨부02] 2025년 11월 게재 예정 KCI등재지 논문

김누리, 양동민. (2025.11).
필수보존속성에 기반한 기록물 디지털화의 제도적 관리 체계 연구.
한국기록관리학회, 25(4).

첨부02	**(게재예정증명서)** 김누리, 양동민. (2025.11). 필수보존속성에 기반한 기록물 디지털화의 제도적 관리 체계 연구. 한국기록관리학회, 25(4).

논문게재예정증명서

한국기록관리학회는 한국기록관리학회지 제25권 제4호에 다음의 논문을 게재할 예정임을 증명합니다.

- 게재예정논문 상세 사항 -

저자명 및 소속기관: 김누리, 양동민 (전북대학교)
게재 학회지명: 한국기록관리학회지 (제25권 제4호)
학회지 발행예정일: 2025년 11월 30일 (제25권 제4호)
게재 논문명: 필수보존속성에 기반한 기록물 디지털화의 제도적 관리 체계 연구

2025. 11. 13.

한 국 기 록 관 리 학 회 장

첨부02	김누리, 양동민. (2025.11). 필수보존속성에 기반한 기록물 디지털화의 제도적 관리 체계 연구. 한국기록관리학회, 25(4).

한국기록관리학회지
25(4), 1-27, 2025.11
https://doi.org/10.14404/JKSARM.2025.25.4.001
pISSN 1598-1487 eISSN 2671-7247

JKSARM
Journal of Korean Society of Archives and Records Management

필수보존속성에 기반한 기록물 디지털화의 제도적 관리 체계 연구*

An Institutional Management Framework for Record Digitization Based on Significant Properties*

김누리(Nuri Kim)[1], 양동민(Dongmin Yang)[2]

E-mail: ri0614@jbnu.ac.kr, dmyang@jbnu.ac.kr

1 제1저자 전북대학교 기록관리학과 석사과정
2 교신저자 전북대학교 기록관리학과 교수, 문화융복합아카이빙연구소 연구원

OPEN ACCESS

논문접수 2025.11.00
최초심사 2025.11.00
게재확정 2025.11.00

ORCID

Nuri Kim
https://orcid.org/0009-0002-7989-8484

Dongmin Yang
https://orcid.org/0000-0002-4029-9372

■ 이 논문은 2023년 대한민국 교육부와 한국연구재단의 인문사회분야 중견연구자지원사업의 지원을 받아 수행된 연구임 (NRF-2023S1A5A2A01077759).
■ 이 연구는 "2025년 행정안전부 국가기록원 기록관리 연구개발사업"의 연구비를 지원받아 수행되었음.

초 록

본 연구는 비전자기록물의 디지털화 과정에서 진본성을 확보하기 위한 제도적인 관리 기준을 마련하고자 하였다. 이를 위해 기록의 본질적인 특성을 규정하는 필수보존속성(Significant Properties) 개념을 도입하고 디지털화 절차를 '계획-실행-검증' 단계로 구조화하였다. 본 연구는 앞선 사례연구(입양기록물·폐교기록물·병적기록물)를 통해 도출된 비전자기록물의 필수보존속성을 종합하여, 디지털화 과정에 적용 가능한 기준으로 재정의하였다. 해외 기록관리기관의 제도를 분석하여 국내 디지털화 제도 개선에 시사점을 제공할 수 있는 요소를 검토하였다. 또한 기록관리전문요원 5인을 대상으로 면담을 실시하여 디지털화 기준의 개선 방향과 본 연구에서 제안한 디지털화 관리 체계와 절차의 가능성을 확인하였다. 본 연구는 디지털화 계획 단계에서 필수보존속성을 정의하고 검증 단계에서 이를 점검 기준으로 활용하는 연계 구조를 제시하였다는 데 의의가 있다. 이러한 제도적 절차의 제안은 디지털화 기록물이 원본과 동등한 지위를 확보하는 데 필요한 이론적 근거로 활용될 수 있다.

ABSTRACT

This study aims to develop an institutional management framework to ensure the authenticity of nondigital records during digitization by establishing clear and actionable criteria. To achieve this, the concept of Significant Properties, which defines the essential characteristics of records, was introduced, and the digitization process was structured into three phases: "planning, digitization, and validation." Based on previous case studies involving adoption, closed university, and military service records, the study redefined the Significant Properties of nondigital records as applicable criteria for digitization. In addition, an analysis of relevant institutional frameworks from overseas archival institutions was conducted to derive insights for improving Korea's digitization framework. Moreover, in-depth interviews with five professional records managers from public institutions were carried out to examine the current limitations and the applicability of the proposed digitization management framework and procedures. The study's significance lies in its structured approach, linking the definition of Significant Properties in the planning phase to their use as evaluation criteria in the validation phase. The proposed institutional procedure may serve as a theoretical foundation for granting digitized records an equivalent status to their originals.

Keywords: 필수보존속성, 디지털화, 비전자기록물, 진본성, 디지털화 절차
Significant Properties, Digitization, Nondigital Records, Authenticity, Digitization Process

https://jksarm.koar.kr

오픈 포맷을 활용한 행정정보 데이터세트
보존 방안 연구

[첨부03] 장기보존패키지 검증 소프트웨어 사용설명서

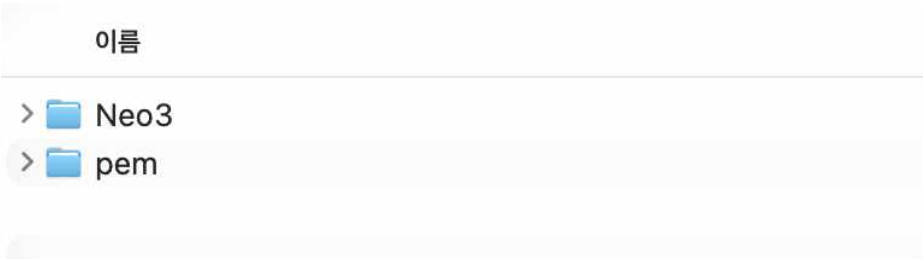
첨3.1 OepnJDK-8 설치

❍ 프로그램을 수행할 장비에서 OpenJDK 17.0.X 버전을 설치한다.(minor 버전은 달라도 무방)

```
bdkim@gimbyeongdong-ui-MacBookPro demo % java -version
openjdk version "17.0.13" 2024-10-15
```

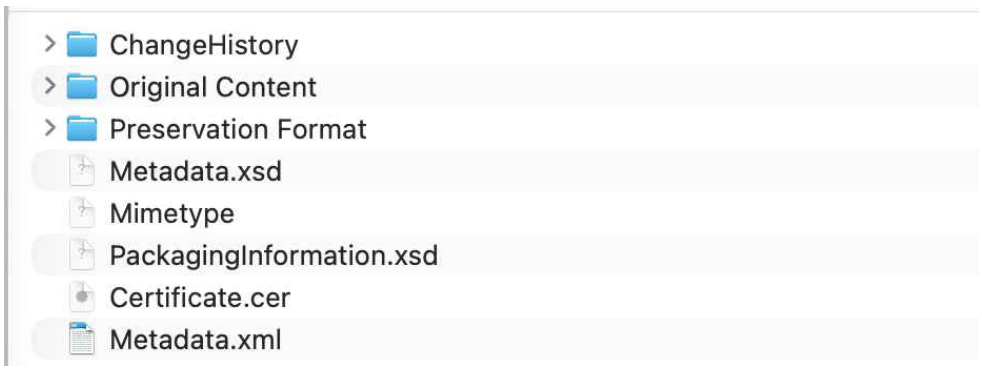
첨3.2 관련 폴더 및 파일 구성

❍ 장기보존패키지 파일들이 있는 폴더명을 “NEO3”로 변경하고 서명 파일이 있는 폴더명을 “pem”으로 변경한다.



❍ 아래와 같이 NEO3 폴더 내에 장기보존패키징 대상이 되는 폴더 및 파일을 구성한다.

폴더(파일)명	구분	용도	생성 방식
ChangeHistory	폴더	History 폴더	이전 버전의 장기보존패키지에 있는 파일들을 복사
Original Content	폴더	원문 폴더	DB의 원문에 해당하는 파일들을 하위에 복사
Preservation Format	폴더	보존포맷 폴더	장기보존이 필요한 보존포맷 파일들을 하위에 복사
Metadata.xsd	파일	장기보존 패키지 Metadata XSD 파일	고정 파일(제공)
Metadata.xml	파일	장기보존 패키지 Metadata XML 파일	시스템에 맞게 생성 필요
PackagingInformation.xsd	파일	생성할 PackagingInformation.xml 에 대한 XSD 파일	고정 파일(제공)
Mimetype	파일	NEO3 Packaging Mimetype	고정 파일(제공)
Certificate.cer	파일	서명 검증을 위한 공개키	GPKI 인증센터에서 발급받은 공개키



[그림] NEO3 폴더 구성

❍ 아래와 같이 pem 폴더 내에 관련 파일들을 구성한다.

폴더(파일)명	구분	용도	생성 방식
Private.key	파일	서명을 위한 개인키	GPKI 인증센터에서 발급받은 개인키
Certificate.cer	파일	서명 검증을 위한 공개키	GPKI 인증센터에서 발급받은 공개키

첨3.3 프로그램 실행

❍ 프로그램을 실행하기 위해서는 기 제공한 Neo3Packager.jar 파일을 특정 폴더에 위치시키고 해당 폴더에서 다음과 같이 명령어를 수행한다.

```
# java -cp Neo3Packager.jar egovframework.neo.utils.NeoWorker $1 $2
$1 : 프로그램 실행 ROOT 폴더(NEO3와 pem폴더가 있는 폴더의 상위 폴더 경로 기술)
$2 : GPKI 인증서 개인키 암호

(실제예시 - root 폴더경로 : /Users/bdkim/java/test/OpenFormat)
java -cp Neo3Packager.jar egovframework.neo.utils.NeoWorker /Users/bdkim/java/test/OpenFormat 1234

(폴더구조 조회)
[bdkim@gimbyeongdong-ui-MacBookPro OpenFormat % cd /Users/bdkim/java/test/OpenFormat
[bdkim@gimbyeongdong-ui-MacBookPro OpenFormat % ls
 Neo3    pem
```

❍ 프로그램이 정상적으로 실행되면 ROOT folder에 다음과 같이 Neo3Packaging.neo3 파일이 정상적으로 생성되는 것을 확인할 수 있다.

```
[bdkim@gimbyeongdong-ui-MacBookPro demo % cd /Users/bdkim/java/test/OpenFormat
[bdkim@gimbyeongdong-ui-MacBookPro OpenFormat % ls
 Neo3                    Neo3Packaging.neo3      pem
[bdkim@gimbyeongdong-ui-MacBookPro OpenFormat % ls -al
```

오픈 포맷을 활용한 행정정보 데이터세트
보존 방안 연구

[첨부04] 데이터세트(DB형) 보존포맷 규격 표준(안)

기록관리 공공표준

NAK xx:20xx(vx.x)

National Archives Standard

데이터세트(DB형) 유형 보존포맷 기술규격(vx.x)

Technical Specification for Dataset(Database-Type) Preservation Format

Version x.0

국가기록원
National Archives of Korea

20xx년 xx월 xx일 제정

○ 제 정 자 : 행정안전부 국가기록원장

○ 제 정 일 : 20xx년 xx월 xx일(국가기록원 고시 제20xx-xx호)

○ 심 의 : 국가기록관리위원회, 기록·정보정책전문위원회

○ 원안작성 :

○ 검 토 :

•

○ 관 리 :

• 국가기록원 정책기획과

(1) 이 표준의 열람은 홈페이지를 이용하시고, 의견 또는 질문은 아래 전화로 연락 주십시오.

• 표준열람 : 국가기록원(http://www.archives.go.kr)→기록관리업무→기록관리표준→표준화현황

• 행정안전부 국가기록원 기록정책부 디지털기록혁신과(042-481-6331)
기록정책부 정책기획과(042-481-6231)

목 차

머리말

이 표준은 장기간에 걸쳐 데이터세트(DB형) 전자기록물의 진본성을 보호하고, 지속적으로 재현 가능하도록 하는데 필요한 보존포맷 규격을 제공하기 위해 제정되었다.

이 표준은 「공공기록물 관리에 관한 법률」 제20조(전자기록물의 관리) 및 동법 시행령 제36조(기록관 및 특수기록관의 전자기록물 보존)에서 규정한 중앙기록물관리기관의 장이 정하는 '보존포맷'의 선정기준에 해당한다

이 표준에서 다루고 있는 '보존포맷'은 「공공기록물 관리에 관한 법률」 제20조(전자기록물의 관리)의 '보존포맷'을 의미하는 것이다. 그리고 보존포맷을 선정하는 기준을 제공하기 위해서 NAK 37「전자기록물 장기보존포맷 기술규격」을, 문서형 보존포맷을 제공하기 위해 NAK 30「문서유형 전자기록물 문서보존포맷 기술규격」을 제정하였다.

이 표준은 DB에서 관리되는 행정정보 데이터세트 전자기록물을 대상으로 하는 보존포맷 규격을 정의한다. 본 데이터세트(DB형) 유형 보존포맷은 진본성을 확보하기 위해 DB의 구조와 내용을 온전히 표현하고, 다양한 시스템 환경에서도 접근성과 가독성이 유지되도록 설계되었다. 또한 재현에 필요한 정보와 무결성 검증 요소를 함께 저장할 수 있도록 구성하였으며, 공공기관이 활용하는 다양한 DBMS 환경에서도 적용 가능하도록 오픈 포맷 기반의 규격을 제시하였다.

이 표준에서 제시하는 데이터세트(DB형) 유형 보존포맷 규격은 국가기록원이 '25년도에 수행한 아래 연구개발사업의 결과물을 기반으로 한다.

- '25년 오픈 포맷을 활용한 행정정보 데이터세트 보존 방안 연구

표준의 구성은 다음과 같다. 제1절부터 제3절까지는 표준의 적용범위, 적용근거 및 용어를 정의하였다. 제4절에서는 보존포맷 다양화 및 데이터세트(DB형) 유형 보존포맷 필요성에 대해 기술하였고, 제5절에서는 데이터세트(DB형) 유형 보존포맷이 포함해야 하는 기록물의 범위와 설계 원칙을 기술하였다. 제6절에서는 데이터세트(DB형) 보존포맷의 폴더 구조와 구성요소에

대한 상세 내용을 설명하였다. 제1절부터 제3절까지는 표준의 적용 범위, 적용 근거, 용어를 정의하였다. 제4절에서는 보존포맷의 다양성과 데이터세트(DB형) 보존포맷의 필요성을 설명하였다. 제5절에서는 해당 보존포맷이 포함해야 할 기록물의 범위와 설계 원칙을 제시하였고, 제6절에서는 데이터세트(DB형) 보존포맷의 폴더 구조와 구성 요소를 상세히 기술하였다.

이 표준은 기록·정보정책전문위원회 및 국가기록관리위원회 심의를 거쳐 제정되었으며, 국가기록원이 유지·관리한다. 이 표준은 관련 법령의 개정, 관계 기관 및 이해당사자의 요청 등 개정사유가 발생할 경우 그 필요성 및 타당성을 검토 후 개정안을 마련하고 의견수렴 및 심의 절차를 거쳐 개정한다.

데이터세트(DB형) 유형 보존포맷 규격

1 적용범위

이 표준은 DB에서 관리되는 데이터세트 전자기록물을 장기간 보존하기 위해 필요한 보존포맷의 기술 규격을 제시한다. 이는 「공공기록물 관리에 관한 법률」과 동법 시행령 규정에 따라 공공기관과 기록물관리기관이 전자기록물을 장기적으로 보존해야 할 때 적용할 수 있는 규격으로, 기록의 진본성·무결성·이용가능성을 안정적으로 유지하는 것을 목표로 한다.

다만, 영구기록물관리기관과 동법 시행령 제3조 각 호에 해당하는 공공기관 등 전자기록물을 장기간 보존하는 기관(이하 '영구기록물관리기관 등'으로 칭함)은 소장 기록물의 특성과 기관의 환경에 따라 데이터세트(DB형) 보존포맷을 달리 정하여 운영할 수 있다.

비고 데이터세트(DB형) 보존포맷은 NAK 37「전자기록물 보존포맷 선정기준」을 참고한다.

2 적용근거

2.1 법적 근거

이 표준의 구체적인 법적 근거는 다음과 같다.

- 「공공기록물 관리에 관한 법률」 제20조(전자기록물의 관리)
- 「공공기록물 관리에 관한 법률」 시행령 제36조(기록관 및 특수기록관의 전자기록물 보존)
- 「공공기록물 관리에 관한 법률」 시행령 제40조(기록관 및 특수기록관의 소관 기록물 이관)
- 「공공기록물 관리에 관한 법률」 시행령 제46조(영구기록물관리기관의 전자기록물 보존 및 관리)

2.2 인용표준

다음의 인용표준은 이 표준의 적용을 위해 필수적이다. 발행연도가 표기된 인용표준은 인용된 판만을 적용한다. 발행연도가 표기되지 않은 인용표준은 최신판(모든 개정내용을 포함)을 적용한다.

- ISO 15489-1:2016, Information and documentation - Records management - Part 1 : Concepts and principles
- ISO 30300(2020) Information and documentation – Records management – Core concepts and vocabulary

2.3 다른 표준과의 연계

이 표준의 적용을 위해 필요하거나 직접적으로 연관이 있는 표준은 다음과 같다. 발행연도가 표기되지 않은 표준은 최신판(모든 개정내용을 포함)을 적용한다.

- KS X 6030 확장가능한 마크업 언어 (XML) - Extensible Markup Language (XML) 1.0
- KS X 6033 확장가능한 마크업 언어 전자서명 구문과 처리 - XML Signature Syntax and Processing
- NAK 30 2022(v1.1) 문서유형 전자기록물 보존포맷 기술규격: PDF/A-1b 기반의 포맷
- NAK 35 2020(v1.0) 행정정보 데이터세트 기록관리 기준-행정정보 데이터세트 기록관리 기준-관리기준표 작성 및 이관규격
- NAK 37 2022(v1.0) 전자기록물 보존포맷 선정기준

3 용어정의

이 표준의 목적을 위해 다음의 용어와 정의를 적용한다.

3.1 기록관리 메타데이터(Metadata for managing records)

시간이 지나가도 영역(domain) 안과 영역 간에 기록의 생산, 관리 및 이용을 가능하게 하는, 구조화되었거나 반구조화된 정보
[ISO 15489-1(2016), ISO 23081-2(2009)]

3.2 보존포맷(Preservation Format)

전자기록물 생산 당시의 내용과 외형 등 주요 특성을 유지함으로써 시간과 기술의 변화와 상관없이 해당 기록물을 재현할 수 있는 포맷

비고 전자기록물의 진본성을 유지하며 장기간 보존하고, 지속적으로 재현할 수 있기 위하여, 전자기록물의 내용, 구조, 맥락, 외관, 기능을 보존할수 있도록 설계된 포맷을 의미한다.

3.3 이용가능성(Useability)

이해관계자들이 합당하다고 여기는 시간 안에 위치를 찾을 수 있고, 검색할 수 있으며, 보여줄 수 있고, 해석할 수 있는 기록의 속성
[ISO 30300(2020), ISO 15489-1(2016) 참조하여 개작]

3.4 장기보존(Long-term Preservation)

오랜 시간이 경과된 후에도 전자기록물에 접근할 수 있고 진본의 상태를 유지하여 증거로서 인정받을 수 있도록 하는 보존행위

3.5 진본성(Authenticity)

기록이 표방하는바 그대로인지, 그것을 생산했거나 보낸 것으로 되어 있는 바로 그 행위주체가 생산했거나 보냈는지, 기록에 명시된 시점에 생산되었거나 보내졌는지를 증명할 수 있는 기록의 품질 [출처 : ISO 30300(2020)]

3.6 필수보존속성(Significant Properties, SP)

시간이 지나도 접근가능하고 의미 있는 상태를 유지하기 위해서 보존되어야 하는 전자기록물의 속성

3.7 데이터(Data)

정보시스템에서 생산, 수집, 가공, 저장, 검색, 제공, 송신, 수신 등을 위해 조

합된 문자, 숫자, 도형, 이미지 및 그 밖의 개체
[「공공기록물 관리에 관한 법률」 시행령 제2조의 내용 개작]

3.8 데이터세트(Dataset)

정보시스템에서 생산, 수집, 가공, 저장, 검색, 제공, 송신, 수신 등을 위해 조합된 문자, 숫자, 도형, 이미지 및 그 밖의 데이터 집합
[「공공기록물 관리에 관한 법률」 시행령 제2조의 11호의 내용 개작]

3.9 데이터베이스(Database)

주어진 목적이나 주어진 자료 처리 시스템에 사용하기에 적합하도록 자료를 구조화하여 자료 검색 및 갱신을 효율화한 자료의 집합
[TTA 정보통신용어사전]

3.10 데이터베이스 관리시스템(Database Management System, DBMS)

데이터베이스(DB)에 접근(access)하여 데이터베이스 정의, 조작, 제어 등 데이터베이스 관리를 지원하는 소프트웨어
[TTA 정보통신용어사전]

3.11 질의-응답 기능(Query-Response)

데이터베이스에서 이용자가 정보를 요청하기 위해 제공한 질의를 데이터베이스가 적용하여 이에 따른 결과를 이용자에게 반환하는 프로세스나 기능

3.12 스키마(Schema)

데이터베이스에 저장되는 데이터의 구조, 제약 조건, 관계 등을 명시적으로 정의하는 체계 또는 도면으로 외부 스키마, 개념 스키마, 내부 스키마의 3중 구조로 이루어짐

3.13 루틴(Routine)

DBMS 내부에 저장되어 필요할 때마다 반복적으로 호출해 사용할 수 있는 프로그램 단위로, 데이터 처리 규칙과 절차를 데이터베이스에서 일관되고 안정적으로 수행하도록 지원

3.14 해시함수(Hash function)

임의의 길이의 문자열을 고정된 길이의 이진 문자열로 매핑하여 주는 함수.

데이터를 자르고, 치환하거나 위치를 바꾸는 방법들로 결과를 만들어 내며, 이 결과를 해시 값(hash value)이라 한다. 해시 함수는 데이터의 무결성, 인증, 부인 방지 등에서 응용되는 중요한 함수 가운데 하나이다.
[TTA 정보통신용어사전]

3.15 확장성 스타일시트 언어 변환(eXtensible Stylesheet Language Transformations, XSLT)

확장성 마크업 언어(XML) 문서를 다른 스타일의 XML 문서로 변경하거나 다른 문서 형식으로 변환하기 위해 개발된 언어. XSLT 기반으로 문서 변환 시 원본 문서는 변경되지 않고 변환 결과로 새로운 문서가 생성된다. HTML, 텍스트(txt), PDF, 포스트스크립트(PostScript), 이미지(PNG 등), 한글(HWP) 등의 형식으로 변환할 수 있다.
[TTA 정보통신용어사전]

4 일반사항

4.1 보존포맷의 다양화 필요성

전자기록물의 장기보존을 위한 보존포맷으로 PDF/A-1(b) 기반의 포맷(NAK 30;2022(v1.1))이 유일한 포맷으로 제공되었으나, 이는 공문서(문서형)에 한정된 단일한 보존포맷으로, 기록물이 생산되는 환경과 기술의 발전으로 공문서도 다양한 보존포맷의 적용이 가능하며, 일반적인 공문서 이외의 전자기록물 유형에 따라 각각 적합한 보존포맷이 필요하다. 따라서 파일포맷의 기술 변화에 대응하기 위해서는 특정 포맷을 지정하기보다, 환경이나 기술의 변화에도 적용이 가능한 보존포맷을 선정하는 기준을 마련하여, 각 공공기관이나 영구기록물관리기관이 자체적으로 적용하도록 하는 체계로의 전환이 필요하다.

행정환경에서는 문서유형 외에 다양한 유형의 전자기록물이 생산되고 있다. 시청각 기록물은 이미지형, 오디오형, 비디오형으로 유형이 세분되며, 유형마다 활용되는 포맷이 각기 다르며, 데이터세트 기록물은 데이터베이스형, 구조화데이터 유형으로 세부되며, 유형에 맞게 활용되는 포맷이 각기 다르다. 문서유형에 비해 이미지형, 오디오형, 비디오형의 시청각기록물과 구조화데

이터와 데이터베이스형의 데이터세트 기록물은 더욱 복잡하고 특수한 보존 기준을 요구한다. 기록물 세부 유형별 맥락, 표현, 기능, 구조 등의 주요 속성들의 '필수보존속성(Significant Properties)' 정의하고, 고유기준에 대한 가중치를 부여하여 고유평가 기준을 마련할 필요가 있다. 또한 전자기록물 유형별 보존포맷 평가표를 작성하여 장기보존에 적정한 다양한 형태의 전자기록물 보존포맷을 선정할 필요가 있다.

4.2 데이터세트(DB형) 유형 보존포맷 필요성

데이터세트(DB형) 전자기록물은 실행 중인 DBMS 안에서만 실시간으로 존재하며, 테이블·스키마·관계정보·루틴·이벤트 등이 복잡하게 얽혀 있어 기존의 하나의 파일포맷으로는 진본성을 유지하기 어렵다. 또한 국내에서는 실제 파일을 DB 내부가 아니라 외부 저장소에 분리해 보관하는 방식이 일반적이어서, 이를 함께 보존해야 데이터세트의 원형 맥락을 유지할 수 있다. 이러한 구조적 특성 때문에 데이터뿐 아니라 DB 고유의 구조와 맥락까지 포괄할 수 있는 별도의 보존포맷이 필요하다.

기존 보존포맷으로 제시되었던 SIARD는 국내 행정정보 데이터세트 관리 기준과 구조적으로 부합하지 않고 지원가능한 DBMS도 제한적이어서 보존포맷으로 적용하기에는 한계가 있다. 또한 SIARD는 전용 소프트웨어 의존도가 높아 공공기관의 실무 활용성이 낮다. 이에 따라 DB형 데이터세트 전자기록물의 장기보존을 위해서는 다양한 DBMS 환경에서 활용할 수 있고 구조적 손실을 최소화할 수 있는 보존포맷이 필수적이다.

데이터세트(DB형) 유형 전자기록물의 장기보존을 위해서는 누구나 사용할 수 있는 개방형 기술(UTF-8, URI, XML 등)을 적용해 데이터의 접근성과 호환성을 높여야 한다. 또한 진본성을 확인하기 위해 보안 및 검증 기술(암호화 해시, XSD 유효성 검사 등)을 활용해 기록이 변조되지 않았음을 확인하고, 구조가 올바르게 유지되도록 관리해야 한다. 더불어 나중에 기록을 재현해야 하는 경우, 데이터 구조와 연계 정보는 물론, 외부 저장소에 보관된 붙임파일까지 함께 포맷 내부에 물리적 또는 논리적으로 포함하는 체계가 필요하다. 이러한 방식으로 데이터를 구조화해 저장해야 자동 처리나 분석에도 활용할 수 있으며, 데이터세트 전체의 맥락을 오래 유지할 수 있다.

5 데이터세트(DB형) 유형 보존포맷 설계

5.1 데이터세트(DB형) 유형 기록물의 범위

데이터세트 유형 전자기록물 보존포맷의 범위는 다음 네 가지 범주로 정의된다.

첫째, **실제 데이터**는 보존포맷의 핵심 구성요소로 반드시 포함된다. 이는 테이블과 레코드 등 DB에 저장된 모든 데이터를 의미하며, 해당 정보시스템에서 수행된 업무 결과를 직접적으로 반영하는 기록의 본질적 요소이다.

둘째, **붙임 파일**은 DB 테이블에는 파일 경로(파일명, URL, 저장 위치 등)만 저장되고, 실제 파일은 외부 스토리지에 보관되는 데이터를 의미한다. 회의록 첨부, 사진, 스캔문서, 증빙자료 등 업무 처리 과정에서 함께 생성·수집된 파일들이 이에 해당한다. 이러한 파일들은 DB와 밀접하게 연계되어 관리되는 정보로서, 실제 데이터와 함께 제시될 때 데이터세트 기록물을 이해하기 위한 중요한 맥락과 근거를 제공한다. 따라서 붙임 파일을 함께 보존해야 기록의 완전성과 진본성을 온전히 확보할 수 있으므로, 필수적으로 포함해야 하는 요소이다.

셋째, **스키마 및 기타요소**는 스키마 구조, 제약조건, 트리거, 프로시저, 뷰 등 DB의 구조와 동작을 정의하는 요소를 말한다. 이러한 요소들은 DBMS 기능이 기록 재현에 필요하다고 판단되는 경우에 선택적으로 포함한다. 특히 데이터의 구조적 의미나 재현 과정에 직접적으로 영향을 미치는 요소들은 보존 대상으로 포함할 수 있으며, 불필요한 경우에는 제외할 수 있다.

넷째, **재현 정보**(이용자 기능·화면)는 데이터가 특정 기능이나 화면을 전제로 설계된 경우, 또는 기능적 맥락이 없으면 의미가 크게 훼손되는 경우에 선택적으로 포함한다. 즉, 데이터 해석이나 활용 과정에서 화면 흐름이나 기능 구조가 필수적인 경우 재현 정보를 함께 보존해야 한다.

이와 같은 보존 범위 구분은 데이터세트(DB형) 유형 보존포맷 설계, 생성 과정에서 일관된 판단 기준으로 활용된다.

5.2 데이터세트(DB형) 유형 보존포맷 설계 원칙

데이터세트(DB형) 유형 전자기록물의 장기보존을 위한 오픈포맷 기반 보존포맷은, 기존 방식의 한계를 보완하고 데이터세트가 지닌 기록적 속성을 온전하게 유지하기 위한 체계적 기준을 수립하는 데 목적이 있다. 본 설계 원칙은 1. 기록의 4대 속성 보장, 2. 기존 SIARD 방식의 한계 해결, 3. 데이터로서의 활용성 확보를 강화하는 관점에서 설정된다.

아래 **표 1**은 데이터세트 유형 보존포맷 설계를 위해 설정한 핵심 원칙을 세 가지 관점에 따라 분류한 것이다.

표 1 - 데이터세트(DB형) 보존포맷 설계 원칙

구분	원칙
1. 기록의 4대 속성 보장	① DB 데이터의 구조와 내용을 온전히 표현할 수 있어야 한다. ② 시스템 환경에 상관없이 데이터의 접근성·가독성·호환성을 확보해야 한다. ③ 재현을 위해 DB 데이터와 밀접하게 연관된 정보를 함께 저장해야 한다. ④ 무결성 정보를 포맷 내부에 내재화해야 한다.
2. 기존 SIARD 방식의 한계 해결	⑤ 데이터세트 단위의 기록관리가 가능해야 한다. ⑥ 국내외 대부분의 DBMS 환경에 적용 가능해야 한다. ⑦ 전용 소프트웨어 없이도 보존포맷을 생성할 수 있어야 한다.
3. 데이터로서의 활용성 확보	⑧ 데이터 처리 및 분석이 용이하도록 구조화된 데이터 형식으로 저장해야 한다.

5.2.1 기록의 4대 속성 보장

① DB 구조와 내용의 온전한 표현: 새로운 보존포맷은 데이터베이스의 구조와 내용을 손실 없이 표현해야 한다. 이를 위해 모든 데이터는 국제 오픈포맷이자 장기보존에 적합한 XML 형식으로 저장하며, 이를 통해 구조적 일관성과 표현의 명확성을 확보한다.

② 환경 독립적 접근성·가독성·호환성 확보: 데이터는 시스템(OS·DBMS) 및 네트워크 환경에 상관없이 접근·열람 가능해야 한다. 이를 위해 보존포맷은 다음을 준수한다.

- UTF-8 인코딩을 기본 인코딩 방식으로 채택
- 내부·외부 파일의 위치 표현은 URI 방식을 사용
- 대용량 첨부가 ZIP(archive)으로 묶여 있을 경우에도 ZIP 내부 파일을 지정할 수 있도록 ZIP 내부 경로 표기 방식(zip!/path)을 허용

예시 보존포맷 attachment/ 폴더에 있는 attach_bundle.zip 내부의 cert/cert_A.hwp 지정
→ attachment/attach_bundle.zip!/cert/cert_A.hwp

③ 재현 가능한 기록으로서의 구성 보존: DB의 데이터만으로는 업무 맥락과 조작 과정이 충분히 재현되지 않는 경우가 존재한다. 이러한 상황을 고려하여, 다음과 같은 구성요소를 필요 시 함께 보존할 수 있도록 설계한다.

- Routine(Trigger, Stored Procedure, Function 등), Event, View, Index, User 정보 등
- 사용자 UI, 기능 등

④ 무결성 검증 정보의 내재화: 데이터의 변조 여부를 확인할 수 있도록 보존포맷 내부에 암호화 해시값(messageDigest)을 필수적으로 내재화한다. 또한 보존포맷이 표준화된 구조로 생성되었음을 검증하기 위해 내부 구조는 XSD(XML Schema Definition)에 따라 검증되도록 한다.

5.2.2 기존 SIARD 방식의 한계 해결

⑤ 데이터세트 단위 기록관리 지원: 기존 SIARD는 DB 전체를 하나의 단위로 묶어 관리하는 구조를 취하여, 테이블보다 더 세분화된 단위의 관리가 어려웠다. SIARD_KR이 테이블 단위 관리를 가능하게 개선되었으나, 국내 기록관리 기준에서 요구하는 '데이터세트' 단위를 지원하기에는 여전히 한계가 있다. 따라서 새로운 보존포맷은 행·열 수준을 넘어 데이터세트 단위의 보다 유연한 관리가 가능하도록 설계한다.

⑥ 다양한 DBMS 적용성 확보: SIARD는 제한된 6종 DBMS만을 지원하여 국내에서 사용되는 국산 DBMS 적용에 제약이 있었다. 새로운 오픈포맷 기반 보존포맷은 국내외 대부분의 DBMS 환경에서도 활용할 수 있는 범

용성을 목표로 설계한다.

⑦ 전용 소프트웨어 의존성 제거: 기존 SIARD 생성에는 전용 도구(SIARD Suite)에 대한 의존성이 존재하였다. 새로운 보존포맷은 전용 소프트웨어 없이 생성할 수 있도록 설계함으로써 공공기관의 활용성을 높인다.

5.2.3 데이터로서의 활용성 확보

⑧ 구조화된 데이터 형식 유지: DB형 데이터세트는 단순히 열람되는 기록이 아니라, 시스템에 의해 자동 처리될 수 있는 데이터로서의 속성을 가진다. 이를 위해 보존포맷의 모든 데이터는 구조화된 데이터(XML 기반)로 저장하여 향후 데이터 처리·분석·연계 활용이 용이하도록 한다.

6 데이터세트(DB형) 유형 보존포맷 폴더 구조와 구성요소

6.1 데이터세트(DB유형) 유형 보존포맷 폴더 구조

그림 1은 데이터세트(DB형) 보존포맷의 기본 폴더 구조를 보여준다. 보존포맷은 metadata.xml, content/table*N*/table*N*.xml, attachment/, representation/, manifest.xml로 구성된다. 이는 각각 실제 데이터, 스키마 및 기타요소, 붙임파일, 재현 정보, 구성목록 정보에 대응된다.

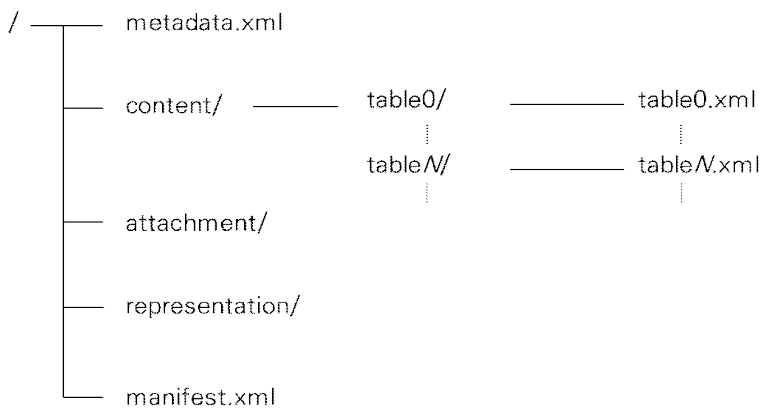


그림 1 - 데이터세트(DB형) 유형 보존포맷 내부 구조

이러한 요소들은 **그림 1**에서 제시된 폴더 구조에 따라 **그림 2**처럼 정해진 위치에 배치된다. 이 구조는 그대로 활용할 수도 있으며, 필요에 따라 전체를 zip 파일로 압축해 보존하는 것도 가능하다.

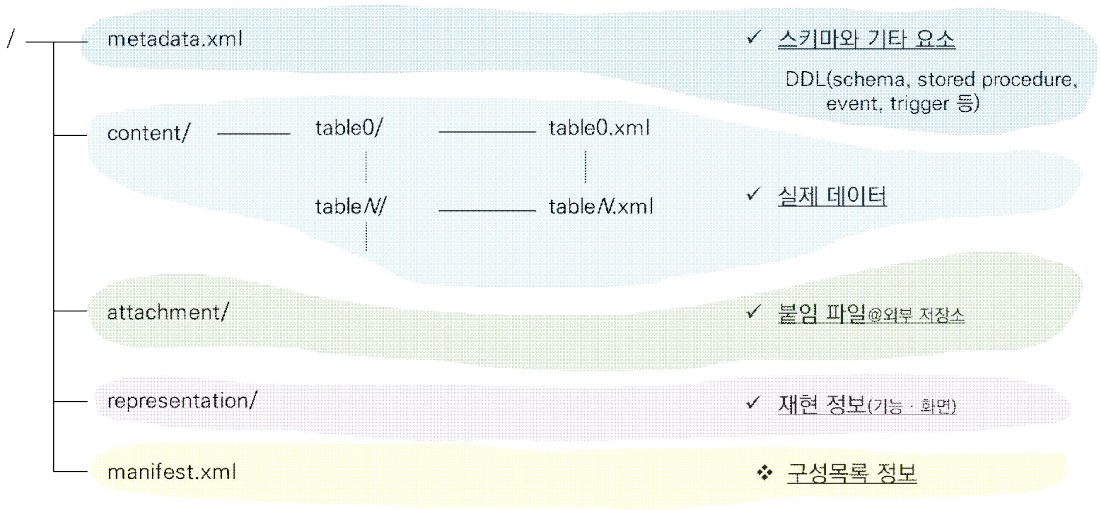


그림 2 - 보존포맷 구성요소와 기록물 보존요소의 대응

최상위 루트 폴더에는 보존포맷 전체를 기술하는 metadata.xml가 위치한다. metadata.xml은 DB의 스키마 및 기타요소를 기술하는 핵심 메타데이터이다. 루트 아래에는 세 개의 하위 폴더가 배치된다.

content/ 폴더는 ㅇ가 저장되는 영역이다. 데이터세트(DB형) 유형의 기록물은 테이블 단위로 저장되며, 각 테이블은 table0/, ..., table*N*/과 같이 개별 폴더로 구분된다. 각 테이블 폴더 안에는 해당 테이블 데이터를 기록한 XML 파일(table0.xml, ..., table*N*.xml)이 포함된다.

attachment/ 폴더는 데이터와 연결된 붙임 파일(회의록 첨부, 사진, 스캔문서, 증빙자료 등)이 보관되는 위치이다.

representation/ 폴더는 재현을 위해 필요한 재현 정보(XSLT, CSS 등) 사용자에게 제공되는 화면, 기능 등과 관련된 파일이 저장되는 위치이다.

manifest.xml는 보존포맷 패키지 내부에 존재하는 모든 파일들을 목록화하는 구성목록 정보이다.

6.2 데이터세트(DB형) 유형 보존포맷의 구성요소

6.2.1 metadata.xml

metadata.xml은 데이터세트(DB형) 보존포맷의 핵심 메타데이터 파일로, 테이블(Table) 정보뿐 아니라 트리거(Trigger), 저장 프로시저(Stored Procedure), 뷰(View), 인덱스(Index) 등 데이터세트 전체를 구성하는 스키마 및 기타 요소에 관한 정보를 포함한다.

metadata.xml의 기본 XML 요소의 구조는 **그림 3**과 같다. <datasets> 요소는 보존포맷의 루트(root)로, 하나의 정보시스템에서 단일 DBMS로 운영되는 데이터세트를 전제로 한 정보를 담는다. 이 요소에는 실제 데이터 스키마를 기술하는 하나 이상의 <dataset> 요소와, 스키마 외의 요소를 표현하기 위한 <routine>, <view>, <event>, <user> 요소가 포함된다. 또한 보존포맷 생성 시점을 나타내는 <archivalDate>와 DBMS의 종류 및 버전을 기록하는 <database> 요소도 함께 구성된다. <datasets>의 요소 설명은 **표 2**와 같다.

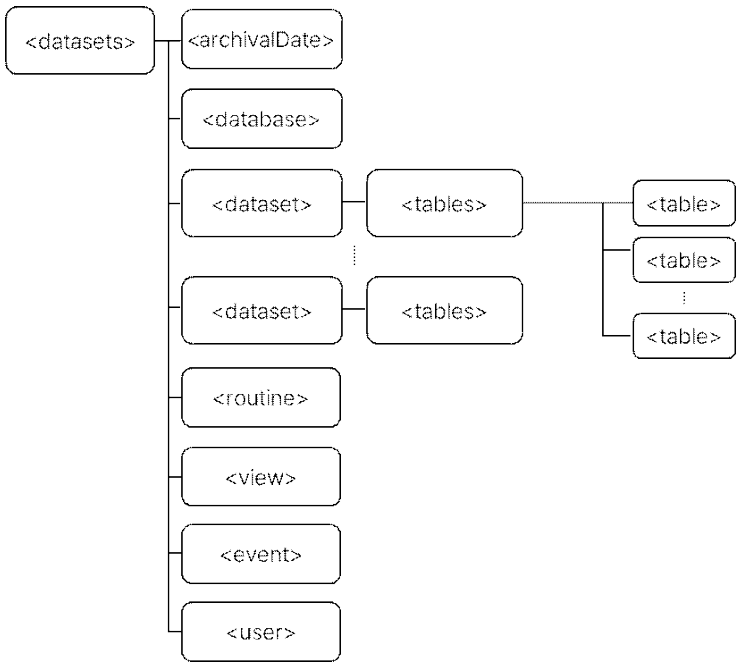


그림 3 - metadata.xml의 기본 XML 요소 구조

표 2 - <datasets> 요소 설명

구분	설명
요소	· **archivalDate** [필수/반복없음] : 보존포맷 생성 시점. ISO 8601 날짜·시간 표기 형식 (예: <archivalDate>2025-11-16T18:06:12</archivalDate>)
	· **database** [필수/반복없음] : DBMS 이름과 버전 (예: <database>MySQL 8.0.18</database>)
	· **dataset** [필수/반복] : 국내 기록관리 기준에서 정의하는 '데이터세트' 단위에 대응함. 데이터세트는 DB 전체가 아니라 특정 '단위 업무'와 관련된 데이터 묶음을 의미하며, 여러 개의 테이블과 선택된 컬럼들로 구성될 수 있음. <dataset>은 하나의 <tables> 요소와 기타 부가 요소로 구성될 수 있음
	· **routine** [필수/반복없음] : stored procedure, function, trigger 등 특정 조건과 입력에 따라 동작을 수행하는 요소를 보관하는 용도로 사용되며, 각 루틴을 생성하는 DDL(Data Definition Language) 명령문을 CDATA 형태로 기록함 (예: <routine> <![CDATA[/*!50003 CREATE*/ /*!50017 DEFINER=`root`@`localhost`*/ /*!50003 TRIGGER `trg_첨부파일_bi` BEFORE INSERT ON `첨부파일` FOR EACH ROW BEGIN SET NEW.첨부파일경로 = CONCAT('C:₩₩Users₩₩박경환₩₩Desktop₩₩대상정보시스템(oasis)₩₩', NEW.첨부파일명); END */;;]]> </routine>)
	· **view** [필수/반복없음] : DBMS에서 사용되는 View 객체를 저장하기 위한 XML 요소임. <tables> 생성 과정에서 View를 함께 확인할 수 있으므로, View 생성과 관련된 DLL 명령어를 CDATA 형태로 이 요소에 기록함 (예: <view> <![CDATA[/*!50001 CREATE VIEW `vw_재직자현황` AS SELECT 1 AS `인사기본ID`, ... 1 AS `성명`, 1 AS `이메일`*/; SET character_set_client = @saved_cs_client;]]> </view>)
	· **event** [필수/반복없음] : DBMS의 Event 객체를 생성하는 DDL 명령어를 CDATA 형태로 기록하기 위한 요소임 (예: <event> <![CDATA[/*!50106 CREATE*/ /*!50117 DEFINER=`root`@`localhost`*/ /*!50106 EVENT `ev_호봉_월간보정` ON SCHEDULE EVERY 1 MONTH STARTS '2025-10-22 00:00:00' ON COMPLETION NOT PRESERVE ENABLE DO BEGIN CALL oasis.sp_호봉_다음승급일_계산(); END */ ;;]]> </event>)

· **user** [필수/반복없음] : 데이터베이스 사용자(User) 계정 및 권한과 관련된 DDL 명령문을 CDATA 형태로 저장하기 위한 요소임
(예: <user> <![CDATA[

```
CREATE TABLE `user` (
 `Host` char(255) CHARACTER SET ascii COLLATE ascii_general_ci NOT
NULL DEFAULT '',
 `User` char(32) COLLATE utf8mb3_bin NOT NULL DEFAULT '',
 utf8mb3_general_ci NOT NULL DEFAULT 'N',
  ...
 `User_attributes` json DEFAULT NULL,
 PRIMARY KEY (`Host`,`User`)
INSERT INTO `user` VALUES
('%','testuser','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N','N',
'N','N','N','N','N','N','N','N','','','','',0,0,0,0,'mysql_native_password','*6BB4837EB743
29105EE4568DDA7DC67ED2CA2AD9','N','2023-11-01
13:14:32',NULL,'N','N','N',NULL,NULL,NULL,NULL),
  ...
)
 ]]>
]]> </user>)
```

metadata.xml은 보존포맷의 **그림 4**와 같이 보존포맷의 다른 요소들과 긴밀하게 연계된다. 예를 들어 <tables> - <table> 요소에는 각 테이블을 생성하는 DDL(Data Definition Language) 명령문이 기록되며, 이를 통해 실제 데이터가 저장된 content/ 폴더와 해당 테이블의 재현 정보를 담는 representation/ 폴더를 연결하는 중심 역할을 수행한다.

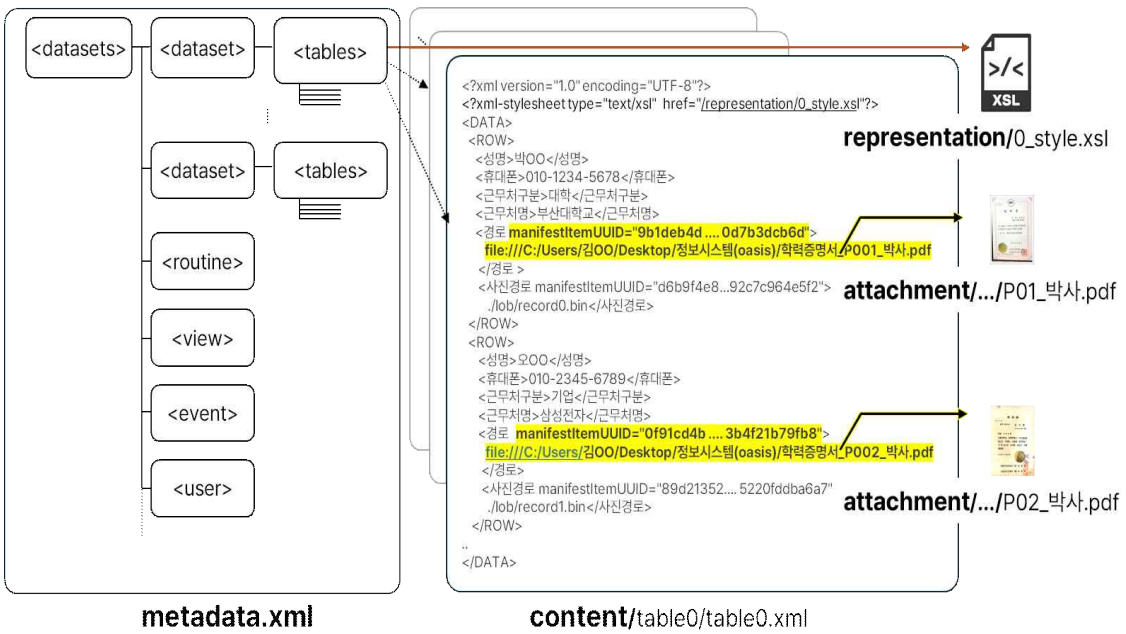


그림 4 - metadata.xml 와 다른 구성요소들과의 관계

다음은 metadata.xml에서 보존포맷의 루트 요소인 <datasets>를 구성하는 하위요소에 대해 설명한다.

1) <dataset> 요소

보존포맷은 국내 행정정보 데이터세트 기록관리 기준에서 정의하는 '단위 업무'를 중심으로 데이터세트를 식별하도록 설계되었다. <dataset> 요소는 이러한 국내 기준의 '데이터세트' 단위와 직접 대응한다.

데이터세트는 DB 전체가 아닌 특정 단위 업무와 관련된 데이터 묶음을 의미하며, 여러 개의 테이블과 선택된 컬럼들로 구성될 수 있다. 또한 생산 형태(DB) 그대로 보존하는 경우 <dataset>은 DB 관리 단위를 의미한다. 예를 들어 Oracle, MySQL, MS SQL Server, Tibero에서는 Schema, 큐브리드(Cubrid)에서는 Database가 이에 해당하며, 모두 <dataset> 요소와 대응된다.

<dataset>은 하나 이상의 <tables> 요소와 필요에 따라 추가되는 기타 부가요소들로 구성된다. 부가 요소는 현재 표준으로 정해져 있지 않으며, 사용자의 필요에 따라 자유롭게 확장할 수 있다. **표 3**은 <dataset>의 요소를 설명한다.

표 3 - <dataset> 요소 설명

구분	설명
속성	· **unitFunctionName** [필수] : 데이터세트 기록관리 대상이 되는 단위기능명을 기술하는 요소로 데이터세트 관리기준표의 '단위기능명' 요소에 기재된 정보를 참조하여 작성 (예: unitFunctionName="신상정보(OAS_02)")
	· **unitFunctionID** [필수] : 단위기능의 식별코드를 기술하는 요소 (예: unitFunctionID="OAS_02")
요소	· **tables** [필수] : <dataset>를 구성하는 데이터들을 포함하고 있는 요소임. 하나 이상의 <table> 요소로 구성
	· **기타 요소** [선택] : · 기타 요소는 현재 표준으로 정하지 않았고, 사용자 정의(User-defined) 방식에 따라 자유롭게 다수의 요소로 확장하여 추가할 수 있음

2) <tables> 요소

<tables> 요소는 여러 개의 <table> 요소로 구성되어 있다. **표 4**는 <tables> 요소를 설명한다.

표 4 - <tables> 요소 설명

구분	설명
요소	· **table** [필수/반복] : <dataset>를 구성하는 데이터들을 포함하고 있는 요소임. 하나 이상의 <table> 요소로 구성

3) <table> 요소

<table> 요소는 하나의 <dataset>을 구성하는 테이블 각각의 생성 방식과 구조를 설명한다. 각 테이블의 데이터는 다음의 두 가지 방식으로 구성될 수 있다.

· 생산형태(DB) 그대로 보존한 데이터
· 서식 형태의 재현을 위해 가공된 데이터

본 연구에서는 위 두 가지 방식의 <table> 구조를 정의하였으며, 그 외 방식은 사용자 정의 방식으로 자유롭게 확장할 수 있도록 설계하였다. **표 5**는 <table> 요소를 설명한다.

표 5 - <table> 요소 설명

요소	내용	
<table> (생성형태 그대로 보존)	**속성**	· **datapath** [필수/반복없음] : 해당 table 요소와 연결된 실제 데이터 파일(tableN.xml)의 절대경로 (예: datapath="/content/table0/table0.xml")
		· **name** [필수/반복없음] : 보존포맷 내 table 명칭. '생성형태 그대로 보존'인 경우 DB의 테이블명 (예: name="개인기본")
		· **sourceType** [필수/반복없음] : 실제 데이터 파일(table*N*.xml)이 생성된 방식. '생성형태 그대로 보존'인 경우, original로 설정 (예: sourceType="original")
	요소	· **sources** [필수/반복없음] : 해당 테이블이 어떤 방식으로 생성되었는지를 설명하는 항목임. '생산형태 그대로 보존'의 경우, 해당 테이블을 생성한 DDL 명령어를

<table>
<tr><td rowspan="5"><table> (서식 형태 보존을 위해 가공)</td><td rowspan="3">속성</td><td>CDATA 형식으로 기록함
(예: <sources> <![CDATA[
CREATE TABLE '개인기본' (
'대표개인번호' varchar(36) COLLATE utf8mb4_unicode_ci NOT NULL,
...
'수정일시' timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP ON
UPDATE CURRENT_TIMESTAMP,
PRIMARY KEY ('대표개인번호'),
UNIQUE KEY 'uk_person_rrn' ('주민등록번호')
)]]> </sources>)</td></tr>
<tr><td>· representationInfo [조건부필수(해당정보 존재시)/반복없음] : 해당 tableN.xml과 연관된 재현 도구 정보. method 속성(xml/xslt, executable, installer, 기타>과 재현 정보에 대한 설명 요소 <description>와 관련 도구들의 위치를 명시하는 <location> 등의 요소로 구성되어 있음
(예: <representation method="xml_xslt">
<description> XML/XSLT를 이용해 table16 데이터를 HTML로 재현하는 방식 </description>
<location>/representation/0_style.xslt</location>
<location>/representation/1_style.css</location>
<location>/content/table/table16.xml</location>
</representation>")</td></tr>
<tr><td>· datapath [필수/반복없음] : 해당 table 요소와 연결된 실제 데이터 파일(tableN.xml)의 절대경로 (예: datapath="/content/table0/table0.xml")</td></tr>
<tr><td>· name [필수/반복없음] : 보존포맷 내 table 명칭. '서식형태 보존을 위해 가공'인 경우 임의로 정함 (예: name="경력증명서")</td></tr>
<tr><td>· sourceType [필수/반복없음] : 실제 데이터 파일(tableN.xml)이 생성된 방식. '서식형태 보존을 위해 가공'인 경우, derived로 설정 (예: sourceType="derived")</td></tr>
<tr><td rowspan="2">요소</td><td>· derivation [필수/반복없음] : 해당 테이블이 도출된 방식을 설명하는 항목임. '서식형태 보존을 위한 가공'인 경우, 해당 테이블을 생성한 SQL 명령어를 CDATA 형식으로 기록함
(예: <derivation><![CDATA[<i>SELECT CONCAT(..... FROM 개인기본 p WHERE p.대표개인번호 = 'P006';</i>]]> </derivation>)</td></tr>
<tr><td>· sources [필수/반복없음] : 해당 테이블을 생성하는 데 사용된 입력 테이블에 대한 설명을 기록함
(예: <sources><![CDATA[
CREATE TABLE '경력' (
'경력ID' bigintNOT NULL AUTO_INCREMENT,
'대표개인번호' varchar(36) COLLATE utf8mb4_unicode_ci NOT NULL,
...
PRIMARY KEY ('경력ID'),</td></tr>
</table>

		KEY '대표개인번호' ('대표개인번호'), *CONSTRAINT '정력_ibfk_2' FOREIGN KEY ('파일ID') REFERENCES '첨부파일' ('파일ID'))* *...* *CREATE TABLE '개인기본' (* *'대표개인번호' varchar(36) COLLATE utf8mb4_unicode_ci NOT NULL,* *...* *'생성일시' timestamp NOT NULL DEFAULT CURRENT_TIMESTAMP,* *PRIMARY KEY ('대표개인번호'),* *KEY 'idx_개인기본_이메일' ('이메일'))]]> </sources>)*
		· **representationInfo** [조건부필수(해당정보 존재시)/반복없음] : 해당 tableN.xml과 연관된 재현 도구 정보. method 속성(xml/xslt, executable, installer, 기타>과 재현 정보에 대한 설명 요소 <description>와 관련 도구들의 위치를 명시하는 <location> 등의 요소로 구성되어 있음 (예: <representation method="xml_xslt"> <description> XML/XSLT를 이용해 table16 데이터를 HTML로 재현하는 방식 </description> <location>/representation/0_style.xslt</location> <location>/representation/1_style.css</location> <location>/content/table/table16.xml</location> </representation>")

6.2.2 content/table*N*/table*N*.xml (*N* = 0, 1, ...)

content/ 폴더는 실제 데이터를 저장하는 공간으로, 각 테이블은 metadata.xml의 <table> 요소에 지정된 datapath 속성과 연결된다. 보존포맷에서는 테이블마다 별도의 table*N*/ 폴더를 생성하고, 그 안에 table*N*.xml 파일을 저장한다. 여기서 *N*은 테이블 번호를 의미하며, 테이블 식별을 위해 0부터 순차적으로 번호를 부여한다(예: DB에서 테이블이 생성된 순서를 기준으로 번호 부여). 데이터 저장 방식은 일반 데이터, LOB 데이터, 붙임 파일 데이터에 따라 서로 다른 방식을 적용하며, 아래 1)~3) 세 가지 방식으로 구성된다.

1) 일반 데이터 저장 방식(행 기반 XML 구조)

일반 데이터는 UTF-8 형식의 '행 기반 XML 구조(row-based XML structure)' 로 table*N*.xml에 저장된다. 데이터베이스의 각 행(row)은 하나의 <ROW> 요소로 표현된다. <ROW> 내부에는 해당 행의 모든 컬럼을 XML

하위 요소로 변환하여 배치하며, 하위 태그 이름은 컬럼명, 태그 내용은 해당 컬럼값이 1:1로 대응된다. 이 방식은 관계형 데이터의 구조를 XML 내에서 직관적이고 일관성 있게 재현할 수 있도록 한다. **그림 5**는 행 기반 XML 구조의 예시를 보여준다.

```
<?xml version="1.0" encoding="utf-8"?>

<DATA>
 <ROW>
  <가족ID>28</가족ID>
  <신상ID>1</신상ID>
  <가족관계구분>배우자</가족관계구분>
  <성명>김은정</성명>
  <주민등록번호>8607152345678</주민등록번호>
  <학력구분>대졸</학력구분>
  <직업_학교명>초등학교교사</직업_학교명>
  <부양가족공제대상여부>1</부양가족공제대상여부>
  <의료보험대상여부>1</의료보험대상여부>
  <가족수당대상여부>1</가족수당대상여부>
  <생성일시>2025-09-19 14:36:26</생성일시>
  <수정일시>2025-09-19 14:36:26</수정일시>
</ROW>
<ROW>
  <가족ID>29</가족ID>
  <신상ID>1</신상ID>
  <가족관계구분>자녀</가족관계구분>
  <성명>박지민</성명>
  <주민등록번호>1205103456789</주민등록번호>
  <학력구분>유치원</학력구분>
  <직업_학교명>햇님유치원</직업_학교명>
  <부양가족공제대상여부>1</부양가족공제대상여부>
  <의료보험대상여부>1</의료보험대상여부>
  <가족수당대상여부>1</가족수당대상여부>
  <생성일시>2025-09-19 14:36:26</생성일시>
  <수정일시>2025-09-19 14:36:26</수정일시>
</ROW>
<ROW>
  <가족ID>30</가족ID>
  ....
```

그림 5 - 행 기반 XML 구조 예시

2) LOB(Large OBject) 데이터 저장 방식

BLOB(Binary LOB), CLOB(Character LOB) 등 LOB 데이터는 다음 두 가지 방식 중 하나로 저장할 수 있다.

첫째, Base64 인코딩 방식(기본 방식)은 LOB 데이터를 Base64로 인코딩하여 table*N*.xml 내부에 직접 포함하는 방식이다. 이 방식은 대부분의 DBMS 애

플리케이션에서 기본적으로 제공하는 기능이므로 특별한 예외 상황이 없다면 기본 방식으로 적용된다.

둘째, lob 폴더 저장 방식은 LOB 데이터를 table*N* 폴더 내부에 별도로 생성한 lob/ 폴더에 저장하는 방식이다. 이 경우 BLOB은 record*M*.bin, CLOB은 record*M*.txt와 같이 생성 순서에 따라 M을 0부터 1씩 증가하면서 파일명을 부여한다. table*N*.xml에는 해당 LOB 파일의 상대경로와 manifestItemUUID를 기록한다. 또한 BLOB 또는 CLOB 파일은 별도의 파일로 저장되므로 manifest.xml에 목록화되며, 이와 함께 해당 LOB의 해시값과 관련 정보가 기록되어 무결성을 검증할 수 있다.

3) 붙임 파일의 데이터 저장 방식

붙임 파일 데이터는 DB에 저장된 값(외부 저장소 경로명)을 table*N*.xml에 그대로 기록한다. 보존포맷 내부에서 해당 첨부파일을 식별하기 위해 manifestItemUUID를 부여하며, manifest.xml에서는 해당 식별자(manifestItemUUID)를 사용하여 파일 경로·해시값·무결성 정보를 관리한다.

6.2.3 attachment/

HTML이나 웹 환경에서는 파일을 직접 열거나 하이퍼링크를 통해 즉시 접근할 수 있지만, LOB 파일은 먼저 파일 형태로 추출해야만 열 수 있어 여러 단계를 거쳐야 하는 불편함이 있다. 이러한 이유로 국내 많은 DB에서는 파일을 LOB 형태로 저장하지 않고, DB 테이블에는 파일의 위치(경로명, URL 등)만 기록한 뒤 실제 파일은 DB 외부의 저장소에 보관한다. 이렇게 외부에 저장된 파일을 '붙임 파일'이라 하며, 보존포맷에서는 이 파일들을 attachment/ 폴더에 포함하여 관리한다.

다만, 파일의 크기나 수량 등 기술적 제약으로 인해 attachment/ 폴더가 아닌 외부 저장소에 둘 필요가 있는 경우도 있다. 이때도 각 파일은 content/table*N*/table*N*.xml 및 manifest.xml과 연계되어야 하므로, content/table*N*/table*N*.xml 파일 내부에 붙임 파일 정보를 기록하는 요소에 manifestItemUUID 속성을 부여해 UUID 방식으로 고유 식별자를 생성하고, 이를 통해 보존포맷 내부에서 추적하고 식별할 수 있도록 한다.

6.2.4 representation/

데이터베이스의 시각적·논리적 구조를 복원하거나 이용자가 화면 형태로 확인할 수 있도록 하기 위해 필요한 재현 정보와 관련된 파일은 representation/ 폴더에 저장된다. 이러한 재현 정보는 특정 테이블과 직접 연결되어야 하므로, metadata.xml의 각 <table> 요소 아래에 <representationInfo> 요소를 두고 재현 도구 파일의 경로를 명시한다.

<representationInfo> 요소는 **표 5**에서 알 수 있듯이 하나의 <description> 요소와 하나 이상의 <location> 요소로 구성되며, 재현 도구가 어떤 방식으로 동작하는지 나타내기 위해 method 속성에 xml_xslt, installer, executable 등의 유형을 지정할 수 있다.

6.2.5 manifest.xml

manifest.xml은 보존포맷을 구성하는 모든 파일을 목록화하여 일괄적으로 관리하는 역할을 한다. 문서의 최상위 요소는 <manifest>이며, 그 하위에 여러 개의 <item> 요소가 포함된다. 각 <item>은 보존포맷 안에 포함된 하나의 파일을 대표한다.

<item> 요소는 hashAlgorithm, createDateTime, manifestItemUUID 속성을 가진다. hashAlgorithm에는 SHA2-256, SHA2-512, SHA-1, MD5 등 파일의 무결성 검증에 사용되는 암호화 해시 알고리즘을 기록한다. createDateTime에는 파일 생성 시점을 ISO 8601 형식으로 저장한다. manifestItemUUID는 보존포맷 내 파일을 식별하기 위한 UUID 기반 고유 식별자로, 별도의 체계 없이도 각 항목을 명확히 구분할 수 있도록 부여된다.

각 <item>은 또한 <hashValue>와 <location> 하위 요소를 갖는다. <hashValue>는 파일의 암호화 해시값을 저장하여, 파일이 훼손되거나 변조되지 않았는지 검증하는 데 사용된다. <location>은 파일의 위치를 URI 형식으로 기록한다. 특히, 파일이 zip 형태의 archive 내부에 있을 경우에도, ‘archive.zip!/inner/path/file.ext’와 같은 zip 내부 경로 표기 방식을 허용하여 개별 파일에 정확히 접근할 수 있도록 한다.

본 연구에서 설계한 보존포맷은 붙임 파일(attachment)을 보존포맷 외부의 저장소에 둘 수 있도록 허용한다. 이 경우 파일 위치가 여러 파일에 중복 기록되면 관리가 복잡해질 수 있으므로, 파일의 실제 위치정보는 manifest.xml에서만 관리하고, 다른 파일에서는 파일을 직접 지칭하지 않고 manifestItemUUID를 통해 연계하는 방식으로 통일하였다. 이를 통해 폴더 구조나 저장 위치가 변경되더라도 manifest.xml 하나만 수정하면 전체 구조가 일관되게 유지될 수 있다. **표 6**은 <manifest>의 하위 요소인 <item>을 설명한다.

표 6 - <manifest> - <item> 요소 설명

구분	설명
속성	· **hashAlgorithm** [필수/반복없음] : 암호화 해시 알고리즘(SHA2-256, SHA2-512, SHA-1, MD5 등) (예: hashAlgorithm="SHA2-512")
	· **createDateTime** [필수/반복없음] : 파일을 생성한 시간. ISO 8601 날짜·시간 표기 형식으로 기록함 (예: createDateTime="2024-04-11T15:10:06")
	· **manifestItemUUID** [필수/반복없음] : 보존포맷을 구성하는 각 파일에 부여되는 고유한 UUID임 (예: manifestItemUUID="9ea34c12-59ff-4e6b-8ac7-2d2a5d4fbb91")
요소	· **hashValue** [필수/반복없음] : hashAlgorithm 속성에 지정된 방식으로 해당 파일에 대해 암호화 해시를 계산한 결과값을 저장함 (예: <hashValue>C3CB8528E570E4B122D3A5EFA6D68286BF42A04B6A1A27CFEA8AA058BDE0FC60DC0E09032714A4DF6201541DDEC182752F857C3FF45366D112416FD4C65B4174 </hashValue>)
	· **location** [필수/반복없음] : 해당 파일의 위치는 기본적으로 URI 방식으로 표시하며, 이는 보존포맷 내부 경로, 외부 저장소 위치, 웹자원 식별자 등을 모두 포함할 수 있음. 필요한 경우에는 zip 파일 내 경로 방식을 활용하여 명시할 수 있음 (예: <location>/content/table0/table0.xml</location> → 보존포맷 내부 파일 경로 <location>/attachment/P001.pdf</location> → 보존포맷 내부 파일 경로 <location>/representation/0_style.xsl</location> → 보존포맷 내부 파일 경로 <location>file:///C:/data/file.pdf</location> → 동일 시스템 내 파일 경로 <location>file:///home/user/data.xml</location> → 동일 시스템 내 파일 경로 <location>https://archive.gov/records/123/file.pdf</location> → 외부 저장소 파일 경로 <location>https://203.245.12.5/api/result.json</location> → 외부 저장소 파일 경로 <location>ftps://192.168.1.100:8080/file/report.pdf</location> → 외부 저장소 파일 경로 <location>urn:isbn:9780306406157</location> → 위치와 상관없이 도서 식별자 <location>/attachment/**a.zip!**/cert/cert_A.hwp</location> → 보존포맷 attachment/ 폴더 아래에 a.zip 파일의 cert/cert_A.hwp를 명시하는 방식 <location> file:///home/user/b.zip!/학력증명서/P001.pdf</location> → 동일 시스템 내 파일 경로. /hom/uesr/b.zip 파일의 학력증명서/P001.pdf를 명시하는 방식)

참고문헌

[1] 국가기록원, 전북대학교 산학협력단. 2019년 11월. 「데이터세트 유형 전자기록의 장기보존기술 연구」.

[2] 국가기록원, 전북대학교 산학협력단. 2020년 11월. 「문서유형 보존포맷 및 장기보존패키지 다양화 연구」.

[3] 국가기록원, 주식회사 그리너리/전북대학교 산학협력단. 2025년 11월 「오픈 포맷을 활용한 행정정보 데이터세트 보존 방안 연구」.

[4] Extensible Markup Language (XML) 1.0 (Fifth Edition), November 10, 2025. Available from Internet <https://www.w3.org/TR/xml/>

[5] ISO/IEC 10646:2020, Information technology — Universal coded character set (UCS),November 10, 2025. Available from Internet <https://www.iso.org/standard/76835.html>

[6] SIARD (Software Independent Archiving of Relational Databases), November 10, 2025. Available from Internet <https://dilcis.eu/content-types/siard>

[7] SIARD_KR, November 10, 2025. Available from Internet <https://github.com/nakdataset/SIARD_KR>

[8] The Extensible Stylesheet Language Family (XSL), November 10, 2025. Available from Internet <https://www.w3.org/Style/XSL/>

[9] Uniform Resource Identifier (URI): Generic Syntax, November 10, 2025. Available from Internet <https://datatracker.ietf.org/doc/html/rfc3986>

[10] UTF-8, a transformation format of ISO 10646, November 10, 2025. Available from Internet <https://datatracker.ietf.org/doc/html/rfc3629>

[11] W3C XML Schema Definition Language (XSD) 1.1, November 10, 2025. Available from Internet <https://www.w3.org/TR/xmlschema11-1/>

총괄 연구과제 요약

과제 고유번호	제1과제		공개가능여부	공개
사업명	2025년 국가기록관리 활용기술 연구개발 사업			
과제명	오픈 포맷을 활용한 행정정보 데이터세트 보존 방안 연구			
연구책임자	성 명	김병동		
	소속 기관명	㈜그리너리		

○ 연구목표 (400 ~ 600자)

❍ 보존포맷 다양화 필요성

- 현재 권장 포맷은 SIARD_KR(국내 환경 맞춤형 도구, CUBRID 지원 등 기능 포함).
- 한계: 지원 DBMS 제한, 버전 호환성 부족, 테이블 단위 이관만 가능(행·열 단위 선택 불가), 사용자 화면 재현 제약.
- 개선 필요하나 지속적 자원 투입이 어려움.
- 따라서 SIARD 외에도 국제표준·오픈포맷 기반의 대체 보존포맷 마련 필요.

❍ 장기보존패키지(NEO3) 규격 보완 필요성

- 국가기록원은 2008년 OAIS 모형과 호주 PROV VEO를 참고해 NEO 표준 개발 → 현재는 ZIP 기반 NEO3 활용.
- NEO3는 다양한 기록유형 수용 가능하나, 여전히 문서 중심 구조에 편중.
- 문서 외 유형(데이터세트 등) 적용 사례 부족, 구성 요소(원문·보존포맷·메타데이터·진본확인)의 구체적 기준 미비.
- 특히 데이터세트 특화 장기보존 메타데이터 정의 부재.
- 따라서 NEO3 개정 또는 데이터세트 전용 장기보존패키지 규격 신설 필요.

➔ 연구목표1 : 행정정보 데이터세트 이관 및 보존하기 위해 SIARD 이외에, 널리 활용되고 국제 표준포맷으로 제정된오픈포맷을 활용한 데이터세트 보존포맷 다양화

➔ 연구목표2 : 현재의 장기보존패키지 NEO3가 행정정보 데이터세트 유형의 전자기록물을 수용할 수 있도록 규격을 전면 개정하거나, 행정정보 데이터세트 유형에 적합한 새로운 장기보존패키지 규격을 제정

○ 연구내용 (1000 ~ 1200자)

❍ 데이터세트 보존포맷 및 장기보존패키지 현황·분석

- 국내외 현황: InfoArchive(OAIS 기반), SIARD2·DBPTK(DB 보존), ETH Data Archive(오픈포맷), MARCXML·EDM(메타데이터 표준) 등 다양한 접근
- 문제점: DBMS 간 호환성 부족, BLOB/CLOB 한계, 비정형 데이터의 포맷 다양성, 대규모 저장 부담
- 대응방안: 국제표준 포맷(SIARD2, RDF/XML, PDF/A, TIFF, WAV) 채택, PID/SID 식별자 체계, 자동 변환·검증(DBPTK) 도입, 하이브리드 저장(S3+ 테이프)
- 기록 외 분야 사례: FAIR 원칙, 버전관리, 선택적 추출·포맷 마이그레이션 활용
- 법제도 분석: 미국 NARA·호주 PROV는 디지털화 시 원본 폐기 근거 명확. Migration은 명시 규정 부족하나 간접 규정 종합 분석 필요

❍ 오픈포맷을 활용한 데이터세트 보존포맷 규격 설계

- 구조 설계: DB형 데이터세트의 구조적 특성을 반영하여 오픈포맷 기반의 보존포맷 설계 원칙과 구조를 제시
- 생성 절차 설계: 메타데이터, 데이터, 첨부파일, 재현파일, 무결성정보로 구성된 보존포맷의 생성 절차를 구체화함

❍ 장기보존패키지(NEO3) 규격 설계

- 구조 정의: 「공공기록물법 시행령」·NAK 31-2 근거, 메타데이터·전자서명·원문 보존포맷 구성
- 보존 범위: DB 데이터+ 붙임은 필수, 재현 정보는 조건부, 주요 산출물은 제외 가능
- 메타데이터 설계: 국가기록원 SIP, 전북대 메타데이터(안) 비교·통합. NAK 8 준용하되 데이터세트 특성 반영 필요
- 기술정보 설계: OAIS 참조모형 기반, DC·EAD·MODS 매핑으로 상호운용성 확보
- 변경이력 관리·복원: 중복 최소화+ 무결성 확보 원칙. '최초 탐색 구성요소 복원 방식'으로 대용량 데이터도 완전 복원 가능

❍ 테스트베드 구축 및 검증

- 재현 시나리오: DB 테이블→XML 변환→XSLT·CSS 적용→HTML 재현. 웹서버 방식·프로그램 방식 병행
- 테스트베드 구성 : (1) 원문·메타데이터 확보 → (2) 폴더 배치 → (3) 파일목록·전자서명·이력관리 → (4) 패키징(.NEO3)
- 검증 시나리오: 전자서명 생성·비교로 진본성 보장

○ 연구성과(응용분야 및 활용범위포함) (400 ~ 600자)

가. 정책활용

· 국가기록원의 데이터세트 기록관리의 장기보존전략 및 정책에 활용
· 국가기록원 장기보존패키지(NEO3) 표준화 확장에 활용
· 영구보존으로 책정된 데이터세트 기록물의 장기보존전략에 활용

나. 타연구/차기연구에 활용

· 시청각기록물, 웹기록물 등 다른 유형의 전자기록물에도 범위를 확장하여 연구

○ 총괄 참여연구원

성 명	소속	성 명	소속
김병동	그리너리	양동민	전북대
김병철	그리너리	김현태	전북대
강하은	그리너리	김지혜	전북대
이동우	그리너리	오경한	전북대
		김누리	전북대
		김 송	전북대
		박라미	전북대
		강지수	전북대
		진엽엽	전북대

Keywords (5개 내외)	*한글*	데이터세트, 보존포맷, 장기보존패키지, 전자기록, 기록물 재현
	영문	Dataset, Preservation Format, Long-Term Preservation Package, Electronic Records, Record Reproduction