국회기록원 지식그래프 구축 안내서

— AP 설계에서 발행까지

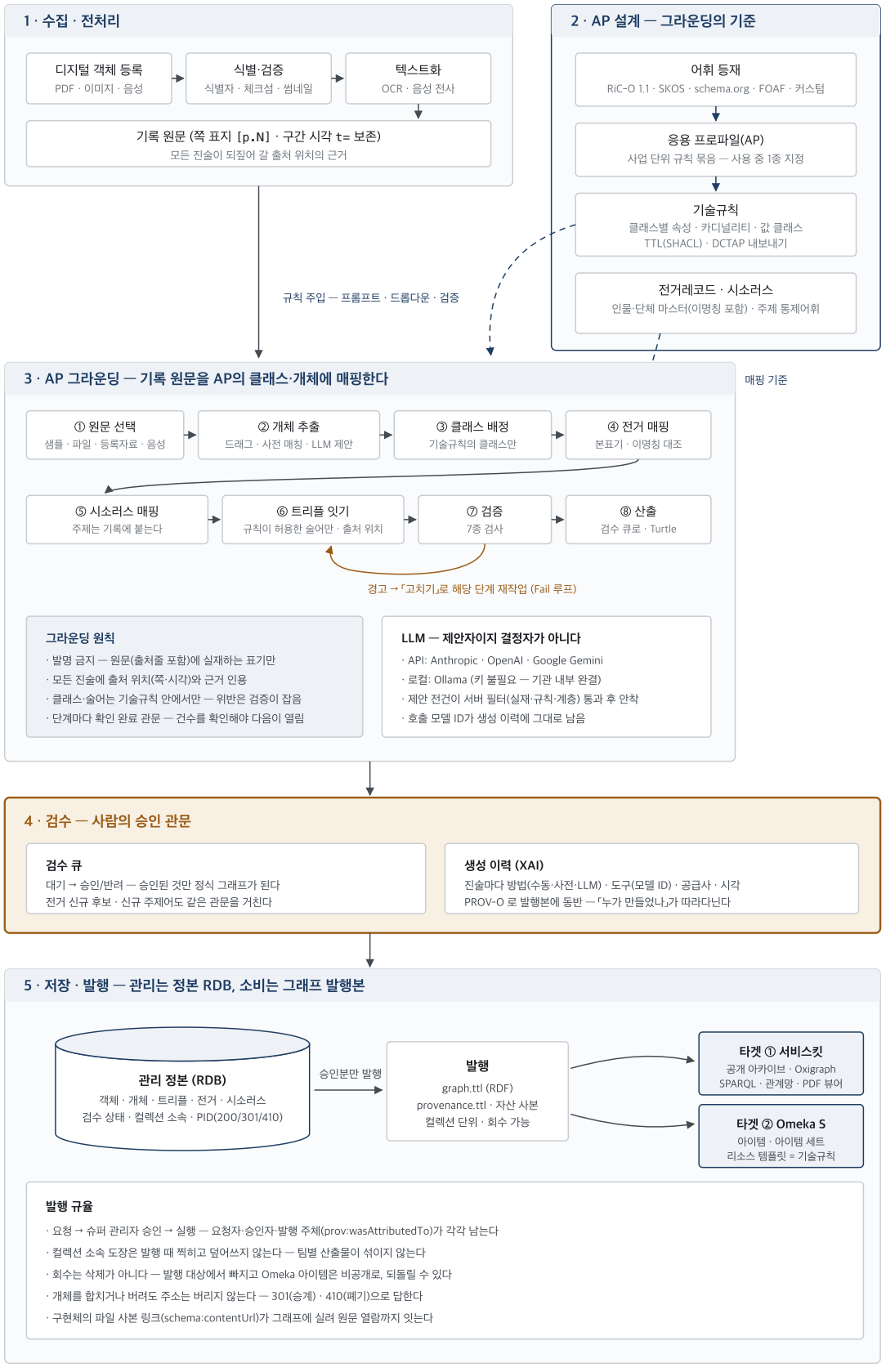

그림 1. 지식그래프 구축 프로세스 — 점선은 AP 설계가 그라운딩·매핑에 기준을 주입하는 흐름.

1. 목적과 배경

구술기록을 비롯한 국회기록원 소장 기록을 **검색 가능한 지식그래프**로 구축하는 표준 프로세스를 정의한다. 핵심 수단은 **응용 프로파일(AP) 설계**와 **AP 그라운딩**이다 — LLM 이 자유롭게 지식을 생성하게 두지 않고, 기관이 미리 설계한 어휘·규칙(AP)에 원문의 표기를 붙들어 매는(grounding) 방식으로, 모든 진술이 **원문 출처와 생성 이력을 동반**하는 그래프를 만든다.

일반적인 OG-RAG 설계가 자동 검증(Validation Pass/Fail)을 통과한 결과를 곧바로 저장하는 것과 달리, 이 프로세스는 검증과 저장 사이에 **사람의 검수 관문**을 하나 더 세워, 승인된 진술만 정식 그래프가 되게 한다. 기록관리 기관의 그래프는 생성 능력보다 **책임 소재**가 앞서야 하기 때문이다.

2. 프로세스 단계

구획	하는 일	산출
1 수집·전처리	디지털 객체를 등록하며 식별자·체크섬·썸네일을 자동 부여하고, OCR·음성 전사로 텍스트화한다. 쪽 표지(`[p.N]`)와 구간 시각(`t=`)을 원문에 보존해 이후 모든 진술의 출처 위치 근거로 삼는다.	출처 위치가 살아 있는 기록 원문
2 AP 설계	표준 어휘(RiC-O 1.1 뼈대 + SKOS·schema.org·FOAF 보강 + 기관 커스텀)를 등재하고, 사업 단위로 응용 프로파일을 묶은 뒤, 클래스별 기술규칙(쓸 속성·카디널리티·값 클래스)을 작성한다. 인물·단체 전거레코드(이명칭 포함)와 주제 시소러스(BT·RT·UF)가 매핑의 마스터가 된다.	기술규칙 (TTL·DCTAP) · 전거 · 시소러스
3 AP 그라운딩	8단계 워크벤치: 원문 선택 → 개체 추출(드래그·사전 매칭·LLM 제안) → 클래스 배정 → 전거 매핑 → 시소러스 매핑(주제는 기록에 붙임) → 트리플 잇기 → 검증 → 산출. 기술규칙이 LLM 프롬프트·화면 드롭다운·검증에 일관되게 주입되고, 검증 경고는 「고치기」로 해당 단계에 되돌아가 재작업하는 Fail 루프를 이룬다. 단계마다 확인 완료 관문이 있어 수행 결과를 건수로 확인해야 다음 단계가 열린다.	출처·근거가 달린 개체·트리플 초안
4 검수	산출물은 곧바로 그래프가 되지 않고 검수 큐에 대기한다. 사람이 승인한 것만 정식 그래프가 되며, 진술마다 생성 방법(수동·사전·LLM)·도구(호출 모델 ID)·공급사·시각이 생성 이력으로 남는다.	승인된 정식 그래프 + 생성 이력
5 저장·발행	정본은 RDB 가 관리하고(검수 상태·컬렉션 소속·PID), 승인분만 RDF(graph.ttl)로 발행해 공개 아카이브(서비스킷)와 Omeka S 두 타겟에 싣는다. 발행은 요청→승인 관문을 거치고, 회수 가능하며, 병합·폐기된 개체의 주소도 301·410 으로 계속 답한다.	공개 지식그래프 · Omeka 아이템

3. 사용 기술

층	기술	역할
온톨로지·어휘	RiC-O 1.1, SKOS, schema.org, FOAF, PROV-O, 기관 커스텀 어휘	기록 기술의 뼈대(RiC-O) · 시소러스(SKOS) · 보강층 · 생성 이력(PROV-O)
규칙 표현	기술규칙 편집기 → SHACL 형 TTL · DCTAP CSV 내보내기	기계와 사람이 같은 규칙을 읽는다 — LLM 프롬프트에도 같은 파일을 준다
지식 추출	LLM API(Anthropic·OpenAI·Google) + 로컬 LLM(Ollama)	개체·관계 후보 제안. 로컬 모델은 키 없이 기관 내부에서 완결 — 외부 반출이 없는 경로
정본 저장	관계형 DB (SQLite — 기관 적용 시 MariaDB/PostgreSQL)	객체·개체·트리플·전거·시소러스를 한 정본에서 — 트랜잭션 · 검수 워크플로 · 감사 이력
그래프 소비	RDF/Turtle 발행 + Oxigraph(브라우저 SPARQL) · Omeka S(JSON-LD)	탐색·질의·시각화는 그래프 저장소가 맡는다 — 정본과 역할 분리
식별 체계	불투명 PID + HTTP 응답 규약(200 정상 · 301 병합 승계 · 410 폐기)	개체를 합치거나 버려도 밖으로 나간 링크가 끊기지 않는다

4. 저장소 구도 — 관리와 소비의 분리

트리플이라고 해서 그래프 DB 에 두어야 하는 것은 아니다. 데이터 모델(그래프)과 저장 기술은 별개이며, 이 프로세스는 **"관리는 RDB 정본, 소비는 그래프 발행본"**으로 역할을 나눈다. 검수·승인·PID 승계 같은 기록관리 워크플로는 트랜잭션이 있는 RDB 가 감당하고, 탐색·SPARQL 질의·관계망 시각화는 발행된 RDF 를 읽는 그래프 저장소가 감당한다. Omeka S 가 내부적으로 MariaDB 에 JSON-LD 값을 쌓는 것과 같은 계열의 패턴이다.

5. 일반적인 OG-RAG 설계와의 비교

일반적인 OG-RAG 설계가 요구하는 **온톨로지 그라운딩 축을 전부 갖춘다**: 데이터 성격에 따른 경로 분리(구조적 레코드는 규칙 매핑, 비정형 원문은 AP 그라운딩), 온톨로지 정의(AP·기술규칙), Entity Linking(전거 매핑), Knowledge Triple 생성, 검증 Fail 루프, 그래프 저장(RDF/Oxigraph)까지 각 구성 요소가 실행 절차에 대응한다. 여기에 통상의 설계에는 없는 **검수 관문과 생성 이력**을 더해, 자동 생성물이 사람의 승인 없이는 그래프가 되지 않게 한 것이 이 프로세스의 고유한 부분이다.

OG-RAG 의 나머지 절반 — 의미론적 청킹·벡터화(Vector DB)·하이브리드 검색·Re-Ranking — 은 이 프로세스의 범위 밖이며, 여기서 구축한 그래프를 근거 저장소로 삼는 **후속 단계**가 된다. 즉 본 프로세스는 OG-RAG 의 「OG」를 먼저 바로 세우는 공정이다.

작성

안대진 · 아카이브랩 | daejin@archivelab.co.kr