좋아, “Verify draft Invoice”를 온톨로지(ontology) 관점으로 깔끔히 잡아볼게. 핵심은 “규칙이 흩어져 있지 않고, 전부 그래프 위에 ‘의미(클래스/속성/제약)’로 올라간다”는 점이야. 그러면 규칙·문서·흐름·승인 근거까지 한 그래프에서 검증·추적·설명 가능해진다.

**1) 개념 스키마(TBox): 무대 세팅**

* **문서 계층**
  + hvdc:Document 상위에 hvdc:Invoice, hvdc:DeliveryOrder, hvdc:StowagePlan, hvdc:WarehouseDoc, hvdc:StatusDoc.
  + 키(식별자)는 hvdc:Key의 서브클래스: hvdc:InvoiceKey, hvdc:DOKey, hvdc:StowageKey, hvdc:WHKey, hvdc:StatusKey, hvdc:ShipmentIDKey, hvdc:BookingKey, hvdc:BLKey, hvdc:ContainerKey 등.  
    → 모든 문서 연결은 이 키들로 그래프 엣지를 만든다.
* **금액/통화**
  + hvdc:MonetaryAmount (액수·통화·단위), hvdc:Currency(예: “USD”, “AED”).
* **레이트/출처**
  + hvdc:RateSource = {Contract, MarketRate, Quotation, SpecialRate} (열거형).
  + hvdc:hasRate, hvdc:hasQuantity, hvdc:hasTotal, hvdc:rateSource.
* **검증 메타**
  + hvdc:VerificationStatus = {VERIFIED, PARTIALLY\_VERIFIED, RATE\_MISMATCH, CURRENCY\_MISMATCH, MULTI\_CURRENCY, REFERENCE\_MISSING, DATA\_MISSING, DOCUMENT\_ALERT, PENDING\_REVIEW}.
  + hvdc:Discrepancy(유형·사유·차이율), hvdc:hasDiscrepancy.
* **흐름/승인/근거(정합성)**
  + 코스트가드 플로우: hvdc:Flow ⟶ hvdc:InvoiceAuditStep (HVDC Logistics Unified v3.7 내 일부).
  + 승인·근거는 **PROV-O** 정렬: prov:Entity(문서), prov:Activity(검증), prov:wasDerivedFrom(참조문서), prov:wasAssociatedWith(담당자).

**2) 제약(Shapes)와 규칙: 그래프 위에서 “검증”을 말로 하지 않고 모델로 한다**

* **SHACL**로 필수 필드, 단위, 포맷을 강제(Invoice/DO/Stowage/WH/Status용 shape). 숫자 필드(레이트·수량)는 0 이상, 소수점 자릿수, 누락 시 DATA\_MISSING, 음수/비수치면 FORMAT\_ERROR.
* **동적 허용오차(레이트 출처별)**
  + Contract: ±3%
  + Market Rate/Quotation: ±5%
  + Special Rate: ±10%
  + 그리고 ±10% 이내는 PENDING\_REVIEW 2차 판정(사람 확인)
  + 합계는 rate × quantity 재계산, 합계 오차 0.01까지 허용
* **통화 규칙**
  + 원문서 통화 유지(환산 금지), 1 USD = 3.6725 AED는 “참고 정보” 어노테이션.
  + 한 인보이스에 다중 통화면 MULTI\_CURRENCY, 참조문서와 통화 다르면 CURRENCY\_MISMATCH.
* **교차문서 일치**
  + 계약/견적/DO 등과 수량·레이트·통화 매칭. 근거 누락·불일치 시 REFERENCE\_MISSING.  
    위 규칙 묶음은 시스템 매뉴얼의 “검증 단계, 상태 코드, 통화 처리, 사전 점검”에 그대로 대응된다.

**3) 워크플로우(그래프 연산 시퀀스)**

1. **사전 점검**: 문서 완전성·통화 일관성·레이트 소스 존재 여부 스캔 → shape 위반나면 즉시 라벨(DATA\_MISSING 등).
2. **콘텐츠 검증**: rate × quantity 재계산, 참조문서 레이트와 비교(출처별 허용오차 반영), 상태 라벨링(VERIFIED/RATE\_MISMATCH/PENDING\_REVIEW…).
3. **교차문서 정합성**: 키로 링크된 계약/견적/DO의 값·승인정보 매칭, 불일치·누락 시 REFERENCE\_MISSING.
4. **요약/리포트 노드**: 총 검증 건수, 상태 분포, 문제 항목 하이라이트(사유 포함).

**4) 그래프 예시(축약 Turtle)**

@prefix hvdc: <https://example.com/hvdc#> .

@prefix prov: <http://www.w3.org/ns/prov#> .

@prefix xsd: <http://www.w3.org/2001/XMLSchema#> .

hvdc:Invoice123 a hvdc:Invoice ;

hvdc:invoiceKey "INV-123" ;

hvdc:currency "AED" ;

hvdc:rateSource hvdc:Contract ;

hvdc:hasRate "150.00"^^xsd:decimal ;

hvdc:hasQuantity "10"^^xsd:decimal ;

hvdc:hasTotal "1500.00"^^xsd:decimal ;

hvdc:references hvdc:Contract789 ;

prov:wasDerivedFrom hvdc:Quotation456 .

# 검증 결과(예)

hvdc:Invoice123\_Validation a hvdc:ValidationActivity ;

hvdc:verificationStatus hvdc:PENDING\_REVIEW ;

hvdc:hasDiscrepancy [

a hvdc:Discrepancy ;

hvdc:discrepancyType hvdc:RateTolerance ;

hvdc:deltaPercent "0.045"^^xsd:decimal ; # 4.5% 차이

] ;

prov:used hvdc:Contract789 ; prov:generated hvdc:Invoice123 .

**5) SHACL로 “규칙=데이터”화(간단 스케치)**

* **필드 존재·형식**: sh:minCount 1, sh:datatype xsd:decimal, sh:minInclusive 0.
* **통화 일관성**: 인보이스 통화와 참조문서 통화 비교(대응 속성에 sh:equals/SPARQL constraints).
* **출처별 허용오차**: SPARQL constraint에서 ?rateSource에 따라 허용오차 분기(Contract 0.03, Market/Quotation 0.05, Special 0.10). ±0.10 이내면 상태를 PENDING\_REVIEW로 마킹.
* **합계 재계산**: 계산식으로 산출값과 제출값의 차이가 0.01 이하인지 확인.

**6) 운영·통합 포인트**

* **키로 연결되는 전사 링크**: hvdc:InvoiceKey 등 키 클래스로 시스템 간 조인 없이 그래프에서 즉시 추론 가능.
* **SCT-EMAIL 매핑**: 메일을 hvdc:Communication(또는 schema:EmailMessage)로 모델링해 승인/합의 근거를 prov:wasDerivedFrom로 인보이스에 귀속.
* **COST-GUARD 플로우**: hvdc:Flow 안에 hvdc:InvoiceAuditStep을 명시해 “어느 단계에서 무슨 규칙으로 걸렸나”를 설명가능하게.
* **명령 ↔ 검증 모드**: /logi-master invoice-audit --deep --highlight-mismatch --ToT\_mode deep
  + --deep: 모든 SHACL shape + SPARQL constraints 전부 실행
  + --highlight-mismatch: hvdc:hasDiscrepancy를 가진 트리플에 태그(또는 리포트에 강조 필드)
  + --ToT\_mode deep: 다단계 규칙(계약→견적→시장가→특수레이트) 체인을 순차 추론
* **ML 연계**: 그래프에서 파생 피처(차이율, 다중통화 여부, 링크 강도, shape 위반 카운트)를 뽑아 이상치 모델에 투입. 모델 결과는 다시 hvdc:AnomalyScore/hvdc:AnomalyFlag로 지식그래프에 적재(설명가능성↑).
* **레포트 스키마**: 리포트도 그래프화(열/요약/상세를 노드로). Excel/대시보드는 그래프 질의 결과의 뷰일 뿐.

원하는 결로 정리하면: \*\*규칙·근거·흐름을 온톨로지로 “고정”\*\*해 두고, SHACL/SPARQL로 검증하고, 키 클래스로 문서를 촘촘히 연결, 통화·출처별 허용오차는 상태코드로 귀결. 그 위에 ML을 얹어 “규칙이 놓치는 패턴”을 보강.  
필요하면 이 스키마/SHACL 초안을 macho715/ontology-insight 스타일에 맞춰 모듈화해서 바로 리포에 붙일 수 있게 만들어줄게.