DRAFT — UNOFFICIAL — NOT FOR OPERATIONAL USE

ARCHITECTURE REFERENCE

ODT-CDA

USAREUR-AF

Architecture Reference

HEADQUARTERS
UNITED STATES ARMY EUROPE AND AFRICA
(USAREUR-AF)
Wiesbaden, Germany

DRAFT — NOT FOR OFFICIAL USE. FOR TRAINING PLANNING PURPOSES ONLY.

20 MARCH 2026

DRAFT — UNOFFICIAL — NOT FOR OPERATIONAL USE

sidebar_position: 2 title: "Engagement Operations"

```
# ============================================================================
# MILITARY OPERATIONS ONTOLOGY - COMPLETE SPECIFICATION
# ============================================================================
# Version: 2.0.0
# Standards: FM 3-0, JP 3-0, ADP 3-0, COW Project, UCDP
# Maintainer: Senior Military Data Scientist
# Last Updated: 2025-11-03
# ============================================================================

ontology:
  version: "2.0.0"
  namespace: "mil.operations"
  metadata:
    cutoff_date: "2025-01-31"
    primary_sources: ["COW", "UCDP", "SIPRI", "NARA", "CGSC"]
    quality_tier: "PRODUCTION"

# ============================================================================
# OBJECT TYPES
# ============================================================================

objectTypes:

  # ----------------------------------------------------------------------------
  # POLITICAL ENTITIES
  # ----------------------------------------------------------------------------

  - apiName: PoliticalEntity
    displayName: Political Entity
    pluralDisplayName: Political Entities
    description: "Any organized polity capable of fielding military forces"
    icon: flag
    primaryKey: entity_id
    titleProperty: name
    properties:
      - apiName: entity_id
        dataType: string
        required: true
        description: "Format: cow_{code} or custom_{slug}"

      - apiName: name
        dataType: string
        required: true
        indexed: true

      - apiName: alternate_names
        dataType: array<string>
        description: "Historical names, translations"

      - apiName: entity_type
        dataType: string
        enum: [
          "SOVEREIGN_NATION",
          "EMPIRE",
          "COLONIAL_TERRITORY",
          "PROTECTORATE",
          "CONFEDERATION",
          "INSURGENCY",
          "SEPARATIST",
          "NON_STATE_ACTOR",
          "PUPPET_STATE",
          "OCCUPATION_GOVERNMENT",
          "PROVISIONAL_GOVERNMENT",
          "CITY_STATE",
          "TRIBAL_CONFEDERATION"
        ]
        required: true

      - apiName: existence_start
        dataType: timestamp
        required: true
        description: "Formation/independence date"

      - apiName: existence_end
```

```
        dataType: timestamp
        description: "Dissolution/conquest. Null = still exists"

      - apiName: cow_code
        dataType: integer
        description: "Correlates of War state code (PRIMARY)"
        indexed: true

      - apiName: iso_code_current
        dataType: string
        description: "ISO 3166-1 alpha-3 if maps to modern state"

      - apiName: fips_code
        dataType: string

      - apiName: gwn_code
        dataType: integer
        description: "Gleditsch & Ward state code"

      - apiName: sovereign_parent_id
        dataType: string
        description: "For non-sovereign entities"

      - apiName: military_autonomy
        dataType: string
        enum: ["FULL", "PARTIAL", "NOMINAL", "NONE"]
        required: true

      - apiName: capital_city
        dataType: string

      - apiName: territory_geojson
        dataType: string
        description: "Borders at existence_start"

      - apiName: population_peak
        dataType: integer

      - apiName: gdp_usd_peak
        dataType: double

      - apiName: data_quality_score
        dataType: integer
        description: "1-10 based on source completeness"

      - apiName: source_citations
        dataType: array<string>
        description: "COW, UCDP, or archival references"

  # ---------------------------------------------------------------------------
  # COALITION
  # ---------------------------------------------------------------------------

  - apiName: Coalition
    displayName: Coalition
    pluralDisplayName: Coalitions
    icon: people-group
    primaryKey: coalition_id
    properties:
      - apiName: coalition_id
        dataType: string
        required: true

      - apiName: name
        dataType: string
        required: true

      - apiName: formation_date
        dataType: timestamp
        required: true

      - apiName: dissolution_date
        dataType: timestamp

      - apiName: formal_treaty
        dataType: boolean

      - apiName: treaty_name
        dataType: string

      - apiName: treaty_text_url
        dataType: string
```

```
      - apiName: unified_command
        dataType: boolean
        description: "NATO vs. ad-hoc coordination"

      - apiName: command_structure
        dataType: string
        enum: ["INTEGRATED", "COORDINATED", "PARALLEL", "AD_HOC"]

  # ---------------------------------------------------------------------------
  # WAR
  # ---------------------------------------------------------------------------

  - apiName: War
    displayName: War
    pluralDisplayName: Wars
    icon: shield-halved
    primaryKey: war_id
    titleProperty: name
    properties:
      - apiName: war_id
        dataType: string
        required: true

      - apiName: name
        dataType: string
        required: true

      - apiName: alternate_names
        dataType: array<string>

      - apiName: start_date
        dataType: timestamp
        required: true

      - apiName: end_date
        dataType: timestamp

      - apiName: war_type
        dataType: string
        enum: [
          "INTERSTATE",
          "CIVIL",
          "COLONIAL",
          "INSURGENCY",
          "INTERVENTION",
          "REGIONAL_CONFLICT"
        ]

      - apiName: geographic_scope
        dataType: string
        enum: ["GLOBAL", "REGIONAL", "BILATERAL", "LOCALIZED"]

      - apiName: strategic_objectives
        dataType: array<string>

      - apiName: total_military_deaths_estimated
        dataType: integer

      - apiName: total_civilian_deaths_estimated
        dataType: integer

      - apiName: outcome
        dataType: string
        enum: [
          "VICTORY_SIDE_A",
          "VICTORY_SIDE_B",
          "STALEMATE",
          "ONGOING",
          "NEGOTIATED_SETTLEMENT"
        ]

      - apiName: cow_war_id
        dataType: integer
        description: "COW War Data ID"

      - apiName: ucdp_conflict_id
        dataType: integer
        description: "UCDP Conflict ID"

      - apiName: data_quality
        dataType: string
        enum: ["HIGH", "MEDIUM", "LOW", "FRAGMENTARY"]
```

```yaml
# ---------------------------------------------------------------------------
# CAMPAIGN
# ---------------------------------------------------------------------------

- apiName: Campaign
  displayName: Campaign
  pluralDisplayName: Campaigns
  icon: map-location-dot
  primaryKey: campaign_id
  properties:
    - apiName: campaign_id
      dataType: string
      required: true

    - apiName: name
      dataType: string
      required: true

    - apiName: theater
      dataType: string
      description: "European, Pacific, CENTCOM AOR, etc."

    - apiName: start_date
      dataType: timestamp
      required: true

    - apiName: end_date
      dataType: timestamp

    - apiName: echelon_level
      dataType: string
      enum: ["THEATER_ARMY", "FIELD_ARMY", "ARMY_GROUP", "CORPS"]

    - apiName: commanding_entity_id
      dataType: string
      description: "PoliticalEntity conducting campaign"

    - apiName: commander_name
      dataType: string

    - apiName: friendly_forces_start_personnel
      dataType: integer

    - apiName: friendly_forces_end_personnel
      dataType: integer

    - apiName: enemy_forces_estimated_start
      dataType: integer

    - apiName: strategic_objectives
      dataType: array<string>

    - apiName: operational_objectives
      dataType: array<string>

    - apiName: outcome
      dataType: string
      enum: ["SUCCESS", "FAILURE", "PARTIAL", "ONGOING"]

    - apiName: geojson_aor
      dataType: string

    - apiName: climate_zone
      dataType: string
      enum: ["ARCTIC", "TEMPERATE", "TROPICAL", "DESERT", "MIXED"]

# ---------------------------------------------------------------------------
# MAJOR OPERATION
# ---------------------------------------------------------------------------

- apiName: MajorOperation
  displayName: Major Operation
  pluralDisplayName: Major Operations
  icon: chess-rook
  primaryKey: operation_id
  properties:
    - apiName: operation_id
      dataType: string
      required: true

    - apiName: name
      dataType: string
      required: true
```

```
    - apiName: operation_type
      dataType: string
      enum: [
        "OFFENSIVE",
        "DEFENSIVE",
        "STABILITY",
        "DEFENSE_SUPPORT_CIVIL_AUTHORITIES",
        "COUNTERINSURGENCY",
        "AMPHIBIOUS",
        "AIRBORNE",
        "COMBINED_ARMS"
      ]
      required: true

    - apiName: start_date
      dataType: timestamp
      required: true

    - apiName: end_date
      dataType: timestamp

    - apiName: echelon_level
      dataType: string
      enum: ["FIELD_ARMY", "ARMY_GROUP", "CORPS", "DIVISION"]

    - apiName: phase_jp_3_0
      dataType: string
      enum: ["PHASE_0", "PHASE_I", "PHASE_II", "PHASE_III", "PHASE_IV", "PHASE_V"]
      description: "Joint Phasing Model"

    - apiName: operational_design
      dataType: string
      enum: [
        "PENETRATION",
        "ENVELOPMENT",
        "TURNING_MOVEMENT",
        "INFILTRATION",
        "FRONTAL_ATTACK",
        "DEFENSE_IN_DEPTH",
        "MOBILE_DEFENSE"
      ]

    - apiName: main_effort_unit_designation
      dataType: string

    - apiName: duration_days_planned
      dataType: double

    - apiName: duration_days_actual
      dataType: double

    - apiName: friendly_casualties_total
      dataType: integer

    - apiName: enemy_casualties_estimated
      dataType: integer

# ------------------------------------------------------------------------
# BATTLE
# ------------------------------------------------------------------------

- apiName: Battle
  displayName: Battle
  pluralDisplayName: Battles
  icon: burst
  primaryKey: battle_id
  titleProperty: name
  properties:
    - apiName: battle_id
      dataType: string
      required: true

    - apiName: name
      dataType: string
      required: true

    - apiName: alternate_names
      dataType: array<string>

    - apiName: start_datetime
      dataType: timestamp
      required: true
```

```
    - apiName: end_datetime
      dataType: timestamp

    - apiName: echelon_level
      dataType: string
      enum: ["CORPS", "DIVISION", "BRIGADE", "REGIMENT"]
      required: true

    - apiName: battle_type
      dataType: string
      enum: [
        "MEETING_ENGAGEMENT",
        "HASTY_ATTACK",
        "DELIBERATE_ATTACK",
        "EXPLOITATION",
        "PURSUIT",
        "DEFENSE",
        "DELAY",
        "WITHDRAWAL",
        "SIEGE",
        "AMPHIBIOUS_ASSAULT",
        "AIRBORNE_ASSAULT"
      ]

    - apiName: friendly_entity_id
      dataType: string
      required: true

    - apiName: friendly_unit_designation
      dataType: string

    - apiName: enemy_entity_id
      dataType: string

    - apiName: enemy_unit_designation
      dataType: string

    # Force Strengths
    - apiName: friendly_strength_start
      dataType: integer
      description: "Personnel"

    - apiName: friendly_strength_end
      dataType: integer

    - apiName: enemy_strength_estimated_start
      dataType: integer

    - apiName: enemy_strength_estimated_end
      dataType: integer

    - apiName: friendly_armor_start
      dataType: integer
      description: "Tanks + IFVs"

    - apiName: friendly_artillery_start
      dataType: integer

    # Casualties
    - apiName: friendly_casualties_kia
      dataType: integer

    - apiName: friendly_casualties_wia
      dataType: integer

    - apiName: friendly_casualties_mia
      dataType: integer

    - apiName: friendly_losses_equipment
      dataType: object
      description: "JSON: {tanks: N, ifv: M, artillery: K, aircraft: J}"

    - apiName: enemy_casualties_estimated_kia
      dataType: integer

    - apiName: enemy_casualties_estimated_wia
      dataType: integer

    - apiName: enemy_losses_equipment_estimated
      dataType: object

    # Environment
```

```
      - apiName: terrain_type
        dataType: string
        enum: [
          "URBAN_DENSE",
          "URBAN_LIGHT",
          "RURAL_OPEN",
          "RURAL_WOODED",
          "MOUNTAIN",
          "DESERT",
          "JUNGLE",
          "LITTORAL",
          "ARCTIC",
          "SWAMP"
        ]

      - apiName: weather_conditions
        dataType: string
        enum: ["CLEAR", "RAIN", "SNOW", "FOG", "EXTREME_HEAT", "EXTREME_COLD"]

      - apiName: visibility_km
        dataType: double

      # Outcomes
      - apiName: objective_achieved
        dataType: boolean

      - apiName: outcome
        dataType: string
        enum: [
          "DECISIVE_VICTORY",
          "VICTORY",
          "MARGINAL_VICTORY",
          "PYRRHIC_VICTORY",
          "STALEMATE",
          "DEFEAT",
          "WITHDRAWAL"
        ]

      - apiName: territorial_change_km2
        dataType: double
        description: "Signed: positive = gained, negative = lost"

      # Geospatial
      - apiName: coordinates_lat
        dataType: double

      - apiName: coordinates_lon
        dataType: double

      - apiName: geojson_battlespace
        dataType: string

      # Sources
      - apiName: source_quality
        dataType: string
        enum: ["PRIMARY", "SECONDARY", "TERTIARY", "ESTIMATED"]

      - apiName: source_citations
        dataType: array<string>

      - apiName: cod_battle_id
        dataType: integer
        description: "Correlates of War Battle ID"

  # ---------------------------------------------------------------------------
  # ENGAGEMENT
  # ---------------------------------------------------------------------------

  - apiName: Engagement
    displayName: Engagement
    pluralDisplayName: Engagements
    icon: crosshairs
    primaryKey: engagement_id
    properties:
      - apiName: engagement_id
        dataType: string
        required: true

      - apiName: name
        dataType: string

      - apiName: start_datetime
        dataType: timestamp
```

```
        required: true

    - apiName: end_datetime
      dataType: timestamp

    - apiName: echelon_level
      dataType: string
      enum: ["BATTALION", "COMPANY", "PLATOON"]
      required: true

    - apiName: engagement_type
      dataType: string
      enum: [
        "MOVEMENT_TO_CONTACT",
        "HASTY_ATTACK",
        "DELIBERATE_ATTACK",
        "DEFEND",
        "RECONNAISSANCE",
        "SECURITY_OPERATION",
        "AMBUSH",
        "RAID",
        "COUNTERATTACK",
        "RETROGRADE"
      ]

    - apiName: friendly_unit_designation
      dataType: string

    - apiName: enemy_unit_designation
      dataType: string

    - apiName: friendly_strength_start
      dataType: integer

    - apiName: friendly_strength_end
      dataType: integer

    - apiName: enemy_strength_estimated_start
      dataType: integer

    - apiName: enemy_strength_estimated_end
      dataType: integer

    - apiName: friendly_casualties_kia
      dataType: integer

    - apiName: friendly_casualties_wia
      dataType: integer

    - apiName: enemy_casualties_estimated
      dataType: integer

    - apiName: fires_support_missions
      dataType: integer

    - apiName: cas_missions
      dataType: integer

    - apiName: objective_description
      dataType: string

    - apiName: objective_achieved
      dataType: boolean

    - apiName: duration_hours
      dataType: double

    - apiName: coordinates_lat
      dataType: double

    - apiName: coordinates_lon
      dataType: double

    - apiName: geojson_location
      dataType: string

# ---------------------------------------------------------------------------
# CONTACT (TIC)
# ---------------------------------------------------------------------------

- apiName: Contact
  displayName: Contact
  pluralDisplayName: Contacts
```

```
  description: "Troops In Contact - tactical level"
  icon: bullseye
  primaryKey: contact_id
  properties:
    - apiName: contact_id
      dataType: string
      required: true

    - apiName: start_datetime
      dataType: timestamp
      required: true

    - apiName: end_datetime
      dataType: timestamp

    - apiName: echelon_level
      dataType: string
      enum: ["PLATOON", "SQUAD", "TEAM", "INDIVIDUAL"]

    - apiName: contact_type
      dataType: string
      enum: [
        "DIRECT_FIRE",
        "INDIRECT_FIRE",
        "IED",
        "VBIED",
        "AMBUSH",
        "SNIPER",
        "COMPLEX_ATTACK",
        "OBSERVATION_ONLY"
      ]

    - apiName: friendly_unit_designation
      dataType: string

    - apiName: friendly_personnel_count
      dataType: integer

    - apiName: enemy_personnel_estimated
      dataType: integer

    - apiName: friendly_casualties
      dataType: integer

    - apiName: enemy_casualties_estimated
      dataType: integer

    - apiName: duration_minutes
      dataType: double

    - apiName: fires_called
      dataType: boolean

    - apiName: medevac_requested
      dataType: boolean

    - apiName: qrf_dispatched
      dataType: boolean

    - apiName: coordinates_lat
      dataType: double

    - apiName: coordinates_lon
      dataType: double

    - apiName: mgrs_grid
      dataType: string

# ---------------------------------------------------------------------------
# FIRE MISSION
# ---------------------------------------------------------------------------

- apiName: FireMission
  displayName: Fire Mission
  pluralDisplayName: Fire Missions
  icon: explosion
  primaryKey: mission_id
  properties:
    - apiName: mission_id
      dataType: string
      required: true

    - apiName: mission_datetime
```

```
        dataType: timestamp
        required: true

      - apiName: mission_type
        dataType: string
        enum: [
          "ARTILLERY_TUBE",
          "ARTILLERY_ROCKET",
          "MORTAR",
          "CAS_FIXED_WING",
          "CAS_ROTARY_WING",
          "STRIKE_FIGHTER",
          "STRIKE_BOMBER",
          "CRUISE_MISSILE",
          "MLRS",
          "NAVAL_GUNFIRE"
        ]
        required: true

      - apiName: firing_unit
        dataType: string

      - apiName: target_type
        dataType: string
        enum: [
          "TROOPS_IN_OPEN",
          "TROOPS_FORTIFIED",
          "ARMOR",
          "ARTILLERY",
          "LOGISTICS_CONVOY",
          "LOGISTICS_DEPOT",
          "COMMAND_POST",
          "AIR_DEFENSE",
          "INFRASTRUCTURE"
        ]

      - apiName: target_description
        dataType: string

      - apiName: rounds_fired
        dataType: integer

      - apiName: munition_type
        dataType: string

      - apiName: munition_weight_kg
        dataType: double

      - apiName: assessed_damage
        dataType: string
        enum: [
          "DESTROYED",
          "SEVERELY_DAMAGED",
          "MODERATELY_DAMAGED",
          "LIGHTLY_DAMAGED",
          "NEAR_MISS",
          "MISS",
          "UNKNOWN"
        ]

      - apiName: coordinates_lat
        dataType: double

      - apiName: coordinates_lon
        dataType: double

      - apiName: cep_meters
        dataType: double
        description: "Circular Error Probable"

  # ---------------------------------------------------------------------------
  # FORCE STRUCTURE
  # ---------------------------------------------------------------------------

  - apiName: MilitaryUnit
    displayName: Military Unit
    pluralDisplayName: Military Units
    icon: users
    primaryKey: unit_id
    properties:
      - apiName: unit_id
        dataType: string
        required: true
```

```
      - apiName: designation
        dataType: string
        required: true
        description: "e.g., '1st Armored Division'"

      - apiName: parent_entity_id
        dataType: string
        required: true

      - apiName: echelon
        dataType: string
        enum: [
          "THEATER_ARMY",
          "FIELD_ARMY",
          "CORPS",
          "DIVISION",
          "BRIGADE",
          "REGIMENT",
          "BATTALION",
          "COMPANY"
        ]
        required: true

      - apiName: unit_type
        dataType: string
        enum: [
          "ARMOR",
          "MECHANIZED_INFANTRY",
          "LIGHT_INFANTRY",
          "AIRBORNE",
          "AIR_ASSAULT",
          "ARTILLERY",
          "AVIATION",
          "ENGINEER",
          "SIGNAL",
          "LOGISTICS",
          "SPECIAL_FORCES"
        ]

      - apiName: formation_date
        dataType: timestamp

      - apiName: dissolution_date
        dataType: timestamp

      - apiName: authorized_strength
        dataType: integer

      - apiName: equipment_major_items
        dataType: object
        description: "JSON MTOE equivalents"

# ============================================================================
# LINK TYPES
# ============================================================================

linkTypes:

  # ----------------------------------------------------------------------------
  # POLITICAL RELATIONSHIPS
  # ----------------------------------------------------------------------------

  - apiName: Succeeds
    displayName: "Succeeds"
    description: "Political entity succession"
    linkTypeA:
      objectType: PoliticalEntity
      cardinality: MANY
      label: "Successor"
    linkTypeB:
      objectType: PoliticalEntity
      cardinality: MANY
      label: "Predecessor"
    properties:
      - apiName: succession_date
        dataType: timestamp
        required: true
      - apiName: succession_type
        dataType: string
        enum: [
          "REVOLUTION",
          "DISSOLUTION",
```

```
          "DECOLONIZATION",
          "UNIFICATION",
          "PARTITION",
          "NAME_CHANGE",
          "REGIME_CHANGE",
          "CONQUEST"
        ]
      - apiName: territorial_continuity_pct
        dataType: double
      - apiName: legal_continuity
        dataType: boolean

- apiName: Colonial
  displayName: "Colonial Relationship"
  linkTypeA:
    objectType: PoliticalEntity
    cardinality: MANY
    label: "Colony"
  linkTypeB:
    objectType: PoliticalEntity
    cardinality: ONE
    label: "Colonizer"
  properties:
    - apiName: start_date
      dataType: timestamp
    - apiName: end_date
      dataType: timestamp
    - apiName: autonomy_level
      dataType: string
      enum: ["NONE", "LIMITED", "DOMINION", "PROTECTORATE"]

- apiName: CoalitionMember
  displayName: "Member of Coalition"
  linkTypeA:
    objectType: PoliticalEntity
    cardinality: MANY
  linkTypeB:
    objectType: Coalition
    cardinality: MANY
  properties:
    - apiName: join_date
      dataType: timestamp
      required: true
    - apiName: leave_date
      dataType: timestamp
    - apiName: membership_status
      dataType: string
      enum: ["FULL_MEMBER", "OBSERVER", "DEFECTED", "EXPELLED"]
    - apiName: command_role
      dataType: string
      enum: ["LEAD_NATION", "MAJOR_CONTRIBUTOR", "MINOR_CONTRIBUTOR", "SUPPORT"]

# ---------------------------------------------------------------------------
# WAR PARTICIPATION
# ---------------------------------------------------------------------------

- apiName: Belligerent
  displayName: "Belligerent In"
  linkTypeA:
    objectType: PoliticalEntity
    cardinality: MANY
  linkTypeB:
    objectType: War
    cardinality: MANY
  properties:
    - apiName: entry_date
      dataType: timestamp
      required: true
    - apiName: exit_date
      dataType: timestamp
    - apiName: side
      dataType: string
      enum: [
        "SIDE_A",
        "SIDE_B",
        "NEUTRAL_LEANING_A",
        "NEUTRAL_LEANING_B"
      ]
      required: true
      description: "Avoid FRIENDLY/HOSTILE - use dyadic coding"
    - apiName: side_change_history
      dataType: array<object>
      description: "JSON: [{date, from_side, to_side, reason}]"
```

```
      - apiName: contribution_type
        dataType: array<string>
      - apiName: peak_troop_contribution
        dataType: integer
      - apiName: total_casualties_kia
        dataType: integer
      - apiName: total_casualties_wia
        dataType: integer
      - apiName: war_expenditure_usd
        dataType: double
      - apiName: fought_as_part_of_entity_id
        dataType: string
        description: "If colonial force under sovereign command"

  - apiName: CoalitionInWar
    displayName: "Coalition Fights In War"
    linkTypeA:
      objectType: Coalition
      cardinality: MANY
    linkTypeB:
      objectType: War
      cardinality: MANY
    properties:
      - apiName: side
        dataType: string
        enum: ["SIDE_A", "SIDE_B"]
        required: true

  # ---------------------------------------------------------------------------
  # OPERATIONAL HIERARCHY
  # ---------------------------------------------------------------------------

  - apiName: PartOfWar
    displayName: "Part of War"
    linkTypeA:
      objectType: Campaign
      cardinality: MANY
    linkTypeB:
      objectType: War
      cardinality: ONE

  - apiName: PartOfCampaign
    displayName: "Part of Campaign"
    linkTypeA:
      objectType: MajorOperation
      cardinality: MANY
    linkTypeB:
      objectType: Campaign
      cardinality: ONE

  - apiName: PartOfMajorOperation
    displayName: "Part of Major Operation"
    linkTypeA:
      objectType: Battle
      cardinality: MANY
    linkTypeB:
      objectType: MajorOperation
      cardinality: ONE

  - apiName: PartOfBattle
    displayName: "Part of Battle"
    linkTypeA:
      objectType: Engagement
      cardinality: MANY
    linkTypeB:
      objectType: Battle
      cardinality: ONE

  - apiName: PartOfEngagement
    displayName: "Part of Engagement"
    linkTypeA:
      objectType: Contact
      cardinality: MANY
    linkTypeB:
      objectType: Engagement
      cardinality: ONE

  # ---------------------------------------------------------------------------
  # FIRES SUPPORT
  # ---------------------------------------------------------------------------

  - apiName: SupportsFire
    displayName: "Supports with Fire"
```

```
    linkTypeA:
      objectType: FireMission
      cardinality: MANY
    linkTypeB:
      objectType: Engagement
      cardinality: ONE
    properties:
      - apiName: time_from_call_to_impact_minutes
        dataType: double
      - apiName: effectiveness_rating
        dataType: integer
        description: "1-5 scale"

  # ---------------------------------------------------------------------------
  # ANALYTICAL LINKS
  # ---------------------------------------------------------------------------

  - apiName: TemporalSuccession
    displayName: "Followed By"
    linkTypeA:
      objectType: Battle
      cardinality: MANY
    linkTypeB:
      objectType: Battle
      cardinality: MANY
    properties:
      - apiName: time_gap_hours
        dataType: double
      - apiName: same_operational_objective
        dataType: boolean

  - apiName: SpatialAdjacency
    displayName: "Adjacent To"
    linkTypeA:
      objectType: Battle
      cardinality: MANY
    linkTypeB:
      objectType: Battle
      cardinality: MANY
    properties:
      - apiName: distance_km
        dataType: double

  - apiName: UnitParticipatedIn
    displayName: "Unit Participated In"
    linkTypeA:
      objectType: MilitaryUnit
      cardinality: MANY
    linkTypeB:
      objectType: Battle
      cardinality: MANY
    properties:
      - apiName: role
        dataType: string
        enum: ["MAIN_EFFORT", "SUPPORTING", "RESERVE", "FLANK_SECURITY"]
      - apiName: casualties_pct
        dataType: double

# ============================================================================
# ACTION TYPES
# ============================================================================

actionTypes:

  - apiName: ComputeAttrition
    displayName: "Compute Attrition"
    appliesTo: [Battle, Engagement]
    parameters:
      - apiName: model_type
        dataType: string
        enum: [
          "LANCHESTER_LINEAR",
          "LANCHESTER_SQUARE",
          "LANCHESTER_LOGARITHMIC",
          "STOCHASTIC_SALVO",
          "SYSTEM_DYNAMICS"
        ]
      - apiName: timestep_hours
        dataType: double
      - apiName: terrain_modifier
        dataType: double
      - apiName: weather_modifier
        dataType: double
```

```
  - apiName: AggregateResults
    displayName: "Aggregate Results"
    appliesTo: [Campaign, MajorOperation, Battle]
    parameters:
      - apiName: aggregation_method
        dataType: string
        enum: ["SUM", "WEIGHTED_AVERAGE", "MONTE_CARLO"]

  - apiName: ValidateData
    displayName: "Validate Data Quality"
    appliesTo: [War, Campaign, Battle, Engagement]
    parameters:
      - apiName: validation_rules
        dataType: array<string>

# ============================================================================
# DERIVED PROPERTIES
# ============================================================================

derivedProperties:

  - objectType: Battle
    apiName: casualty_exchange_ratio
    dataType: double
    formula: "NULLIF(friendly_casualties_kia, 0) / NULLIF(enemy_casualties_estimated_kia, 1)"
    description: "Avoid division by zero"

  - objectType: Battle
    apiName: combat_power_remaining_pct
    dataType: double
    formula: "(friendly_strength_end / NULLIF(friendly_strength_start, 1)) * 100"

  - objectType: Battle
    apiName: duration_days
    dataType: double
    formula: "EXTRACT(EPOCH FROM (end_datetime - start_datetime)) / 86400"

  - objectType: Engagement
    apiName: attrition_rate_per_hour
    dataType: double
    formula: "friendly_casualties_kia / NULLIF(duration_hours, 1)"

  - objectType: War
    apiName: total_deaths
    dataType: integer
    formula: "total_military_deaths_estimated + total_civilian_deaths_estimated"
```


==========

DATA INGESTION PLAN

1. SOURCE DATA MAPPING

```
SOURCE_REGISTRY = {
    "PoliticalEntity": [
        {
            "source": "COW State System",
            "url": "https://correlatesofwar.org/data-sets/state-system-membership/",
            "format": "CSV",
            "coverage": "1816-2016",
            "update_frequency": "Annual",
            "priority": "PRIMARY",
            "quality": 9,
            "fields_mapped": [
                "cow_code", "name", "existence_start", "existence_end",
                "entity_type=SOVEREIGN_NATION"
            ],
            "validation": "COW coding rules apply"
        },
        {
            "source": "Gleditsch & Ward States",
            "url": "http://ksgleditsch.com/data-4.html",
            "format": "DTA/CSV",
            "coverage": "1816-2017",
            "priority": "SECONDARY",
            "quality": 9,
            "fields_mapped": ["gwn_code", "alternate_names"],
            "merge_strategy": "LEFT JOIN on cow_code"
        },
        {
            "source": "Historical State Systems (Custom)",
            "url": "NARA/Archival",
            "format": "Manual Entry",
            "coverage": "Pre-1816, non-state actors",
            "priority": "TERTIARY",
            "quality": 6,
            "fields_mapped": [
                "entity_type IN (EMPIRE, COLONIAL_TERRITORY, INSURGENCY, etc.)"
            ],
            "notes": "Requires SME review for Ottoman Empire, HRE, etc."
        }
    ],

    "War": [
        {
            "source": "COW Inter-State War Data",
            "url": "https://correlatesofwar.org/data-sets/cow-war/",
            "format": "CSV",
            "coverage": "1816-2007",
            "version": "5.0",
            "priority": "PRIMARY",
            "quality": 9,
            "fields_mapped": [
                "cow_war_id", "name", "start_date", "end_date",
                "war_type=INTERSTATE", "total_military_deaths_estimated"
            ]
        },
        {
            "source": "COW Intra-State War Data",
            "url": "https://correlatesofwar.org/data-sets/cow-war/",
            "format": "CSV",
            "coverage": "1816-2014",
            "version": "5.1",
            "priority": "PRIMARY",
            "quality": 8,
            "fields_mapped": [
                "war_type=CIVIL", "belligerents via side_a/side_b"
            ]
        },
        {
            "source": "UCDP Armed Conflict Dataset",
            "url": "https://ucdp.uu.se/downloads/",
            "format": "CSV",
            "coverage": "1946-2023",
            "priority": "PRIMARY",
```

```
            "quality": 9,
            "fields_mapped": [
                "ucdp_conflict_id", "best_fatality_estimate",
                "geographic_scope via intensity"
            ],
            "merge_strategy": "MATCH on war_name + date range"
        },
        {
            "source": "PRIO Battle Deaths",
            "url": "https://www.prio.org/data/1",
            "format": "CSV",
            "coverage": "1946-2008",
            "priority": "VALIDATION",
            "quality": 8,
            "use_case": "Cross-check casualty estimates"
        }
    ],

    "Battle": [
        {
            "source": "COW Directed Dyadic MID",
            "url": "https://correlatesofwar.org/data-sets/mids/",
            "format": "CSV",
            "coverage": "1816-2014",
            "priority": "SECONDARY",
            "quality": 7,
            "notes": "MIDs include battles; filter hostility_level >= 4"
        },
        {
            "source": "Clodfelter Historical Battles",
            "citation": "Clodfelter, M. (2017). Warfare and Armed Conflicts",
            "format": "Book/Manual Entry",
            "coverage": "3000 BCE - 2015 CE",
            "priority": "TERTIARY",
            "quality": 7,
            "fields_mapped": [
                "name", "date", "participants", "casualties (estimated)",
                "outcome (qualitative)"
            ],
            "notes": "Gold standard for pre-20th century. Manual digitization required."
        },
        {
            "source": "DoD Operational Reports (OPSREPS)",
            "format": "XML/PDF",
            "coverage": "Iraq/Afghanistan 2001-2021",
            "classification": "CUI",
            "priority": "PRIMARY (modern)",
            "quality": 9,
            "access": "Requires CAC/clearance"
        },
        {
            "source": "Army Battle Command System (ABCS) Logs",
            "format": "Proprietary/CSV export",
            "coverage": "OIF/OEF selected units",
            "classification": "SECRET",
            "priority": "PRIMARY (modern)",
            "quality": 10,
            "fields_mapped": "ALL battle-level properties at engagement granularity"
        }
    ],

    "Engagement": [
        {
            "source": "SIGACT Database",
            "format": "SQL dump",
            "coverage": "Afghanistan 2001-2014, Iraq 2003-2011",
            "classification": "SECRET//REL",
            "priority": "PRIMARY",
            "quality": 8,
            "fields_mapped": [
                "Contact-level events aggregated to Engagement",
                "TIC reports", "casualty data", "geospatial"
            ]
        }
    ],

    "FireMission": [
        {
            "source": "AFATDS Mission Archives",
            "format": "Database export",
            "coverage": "Theater-dependent",
            "classification": "SECRET",
            "priority": "PRIMARY",
```

```
            "quality": 10
        }
    ],

    "Coalition": [
        {
            "source": "NATO Membership",
            "url": "https://www.nato.int/cps/en/natolive/topics_52044.htm",
            "format": "Web scrape/Manual",
            "coverage": "1949-present",
            "priority": "PRIMARY",
            "quality": 10
        },
        {
            "source": "UN Peacekeeping Missions",
            "url": "https://peacekeeping.un.org/en/data",
            "format": "CSV",
            "coverage": "1948-present",
            "priority": "PRIMARY",
            "quality": 9
        }
    ]
}
```

2. ETL PIPELINE ARCHITECTURE

```
"""
Production ETL Pipeline for Military Operations Ontology
Uses: DuckDB, dlt, Pydantic validation, Ruff compliance
"""

from dataclasses import dataclass
from datetime import datetime
from pathlib import Path
from typing import Any, Dict, List, Optional
import duckdb
import dlt
from pydantic import BaseModel, Field, validator


# ============================================================================
# CONFIGURATION
# ============================================================================

@dataclass
class PipelineConfig:
    """Pipeline configuration."""

    source_dir: Path = Path("/mnt/data/sources")
    staging_dir: Path = Path("/mnt/data/staging")
    output_dir: Path = Path("/mnt/user-data/outputs")
    duckdb_path: Path = Path("/mnt/data/ontology.duckdb")

    cow_states_url: str = "https://correlatesofwar.org/wp-content/uploads/system2016.csv"
    cow_wars_url: str = "https://correlatesofwar.org/wp-content/uploads/Inter-StateWarData_v4.0.csv"
    ucdp_url: str = "https://ucdp.uu.se/downloads/ucdpprio/ucdp-prio-acd-231.csv"

    log_level: str = "INFO"
    validate_strict: bool = True
    quality_threshold: int = 6


# ============================================================================
# PYDANTIC MODELS (DATA VALIDATION)
# ============================================================================

class PoliticalEntityModel(BaseModel):
    """Validation for PoliticalEntity."""

    entity_id: str
    name: str
    entity_type: str
    existence_start: datetime
    existence_end: Optional[datetime] = None
    cow_code: Optional[int] = None
    iso_code_current: Optional[str] = None
    military_autonomy: str = "FULL"
```

```
    data_quality_score: int = Field(ge=1, le=10)

    @validator("entity_type")
    def validate_entity_type(cls, v):
        allowed = [
            "SOVEREIGN_NATION", "EMPIRE", "COLONIAL_TERRITORY",
            "PROTECTORATE", "CONFEDERATION", "INSURGENCY", "SEPARATIST",
            "NON_STATE_ACTOR", "PUPPET_STATE", "OCCUPATION_GOVERNMENT",
            "PROVISIONAL_GOVERNMENT", "CITY_STATE", "TRIBAL_CONFEDERATION"
        ]
        if v not in allowed:
            raise ValueError(f"Invalid entity_type: {v}")
        return v

    @validator("military_autonomy")
    def validate_autonomy(cls, v):
        if v not in ["FULL", "PARTIAL", "NOMINAL", "NONE"]:
            raise ValueError(f"Invalid military_autonomy: {v}")
        return v


class WarModel(BaseModel):
    """Validation for War."""

    war_id: str
    name: str
    start_date: datetime
    end_date: Optional[datetime] = None
    war_type: str
    outcome: str = "ONGOING"
    total_military_deaths_estimated: Optional[int] = Field(ge=0, default=None)
    cow_war_id: Optional[int] = None
    ucdp_conflict_id: Optional[int] = None
    data_quality: str = "MEDIUM"

    @validator("war_type")
    def validate_war_type(cls, v):
        allowed = ["INTERSTATE", "CIVIL", "COLONIAL", "INSURGENCY",
                   "INTERVENTION", "REGIONAL_CONFLICT"]
        if v not in allowed:
            raise ValueError(f"Invalid war_type: {v}")
        return v


class BattleModel(BaseModel):
    """Validation for Battle."""

    battle_id: str
    name: str
    start_datetime: datetime
    echelon_level: str
    friendly_entity_id: str

    friendly_strength_start: Optional[int] = Field(ge=0, default=None)
    friendly_casualties_kia: Optional[int] = Field(ge=0, default=None)

    outcome: Optional[str] = None
    source_quality: str = "SECONDARY"

    @validator("echelon_level")
    def validate_echelon(cls, v):
        if v not in ["CORPS", "DIVISION", "BRIGADE", "REGIMENT"]:
            raise ValueError(f"Invalid echelon_level: {v}")
        return v

    @validator("friendly_casualties_kia")
    def validate_casualties(cls, v, values):
        """Casualties cannot exceed starting strength."""
        if v and "friendly_strength_start" in values:
            start = values["friendly_strength_start"]
            if start and v > start:
                raise ValueError("Casualties exceed starting strength")
        return v


# ============================================================================
# EXTRACTION FUNCTIONS
# ============================================================================

def extract_cow_states(config: PipelineConfig) -> List[Dict[str, Any]]:
    """Extract COW State System data.

    Returns:
```

```
        List of raw political entity records.
    """
    import pandas as pd

    print(f"Extracting COW states from {config.cow_states_url}")

    df = pd.read_csv(config.cow_states_url)

    records = []
    for _, row in df.iterrows():
        records.append({
            "entity_id": f"cow_{row['ccode']}",
            "name": row["statenme"],
            "entity_type": "SOVEREIGN_NATION",
            "existence_start": f"{row['styear']}-01-01",
            "existence_end": f"{row['endyear']}-12-31" if row["endyear"] < 2016 else None,
            "cow_code": int(row["ccode"]),
            "military_autonomy": "FULL",
            "data_quality_score": 9,
            "source_citations": ["COW State System v2016"]
        })

    print(f"Extracted {len(records)} political entities")
    return records


def extract_cow_wars(config: PipelineConfig) -> List[Dict[str, Any]]:
    """Extract COW Inter-State War data."""
    import pandas as pd

    print(f"Extracting COW wars from {config.cow_wars_url}")

    df = pd.read_csv(config.cow_wars_url)

    records = []
    for _, row in df.iterrows():
        records.append({
            "war_id": f"cow_war_{row['WarNum']}",
            "name": row["WarName"],
            "start_date": f"{row['StartYear']}-{row['StartMonth1']:02d}-{row['StartDay1']:02d}",
            "end_date": f"{row['EndYear']}-{row['EndMonth1']:02d}-{row['EndDay1']:02d}",
            "war_type": "INTERSTATE",
            "outcome": "VICTORY_SIDE_A" if row["Outcome"] == 1 else "STALEMATE",
            "total_military_deaths_estimated": int(row["BatDeath"]) if pd.notna(row["BatDeath"]) else None,
            "cow_war_id": int(row["WarNum"]),
            "data_quality": "HIGH",
            "source_citations": ["COW Inter-State War v5.0"]
        })

    print(f"Extracted {len(records)} wars")
    return records


def extract_belligerent_links(config: PipelineConfig) -> List[Dict[str, Any]]:
    """Extract war participation links."""
    import pandas as pd

    df = pd.read_csv(config.cow_wars_url)

    links = []
    for _, row in df.iterrows():
        war_id = f"cow_war_{row['WarNum']}"

        # Side A participants
        if pd.notna(row["StateName"]):
            links.append({
                "source_id": f"cow_{row['ccode']}",
                "target_id": war_id,
                "entry_date": f"{row['StartYear']}-{row['StartMonth1']:02d}-{row['StartDay1']:02d}",
                "exit_date": f"{row['EndYear']}-{row['EndMonth1']:02d}-{row['EndDay1']:02d}",
                "side": "SIDE_A",
                "total_casualties_kia": int(row["BatDeath"]) if pd.notna(row["BatDeath"]) else None
            })

    print(f"Extracted {len(links)} belligerent relationships")
    return links


# ============================================================================
# TRANSFORMATION & VALIDATION
# ============================================================================

def validate_and_transform(
```

```
    records: List[Dict[str, Any]],
    model: type[BaseModel],
    config: PipelineConfig
) -> List[Dict[str, Any]]:
    """Validate records using Pydantic models.

    Args:
        records: Raw extracted records.
        model: Pydantic model class for validation.
        config: Pipeline configuration.

    Returns:
        Validated and cleaned records.
    """
    validated = []
    errors = []

    for i, record in enumerate(records):
        try:
            validated_record = model(**record)
            validated.append(validated_record.dict())
        except Exception as e:
            errors.append({"record_index": i, "error": str(e), "record": record})
            if config.validate_strict:
                raise

    if errors:
        print(f"WARNING: {len(errors)} validation errors")
        # Log errors to file
        error_path = config.staging_dir / "validation_errors.json"
        import json
        with open(error_path, "w") as f:
            json.dump(errors, f, indent=2)

    print(f"Validated {len(validated)}/{len(records)} records")
    return validated


# ============================================================================
# LOADING TO DUCKDB
# ============================================================================

def load_to_duckdb(
    table_name: str,
    records: List[Dict[str, Any]],
    config: PipelineConfig
) -> None:
    """Load validated records to DuckDB.

    Args:
        table_name: Target table name.
        records: Validated records.
        config: Pipeline configuration.
    """
    conn = duckdb.connect(str(config.duckdb_path))

    # Create table if not exists
    conn.execute(f"""
        CREATE TABLE IF NOT EXISTS {table_name} AS
        SELECT * FROM read_json_auto(?)
        WHERE 1=0
    """, [records])

    # Insert records
    conn.execute(f"""
        INSERT INTO {table_name}
        SELECT * FROM read_json_auto(?)
    """, [records])

    row_count = conn.execute(f"SELECT COUNT(*) FROM {table_name}").fetchone()[0]
    print(f"Loaded {len(records)} records to {table_name} (total: {row_count})")

    conn.close()


# ============================================================================
# MAIN PIPELINE
# ============================================================================

def run_pipeline() -> None:
    """Execute full ETL pipeline."""
    config = PipelineConfig()
```

```
    # Ensure directories exist
    config.staging_dir.mkdir(parents=True, exist_ok=True)
    config.output_dir.mkdir(parents=True, exist_ok=True)

    print("=" * 80)
    print("MILITARY OPERATIONS ONTOLOGY ETL PIPELINE")
    print("=" * 80)

    # ------------------------------------------------------------------------
    # PHASE 1: POLITICAL ENTITIES
    # ------------------------------------------------------------------------
    print("\n[1/4] Extracting Political Entities...")
    entities_raw = extract_cow_states(config)

    print("[1/4] Validating Political Entities...")
    entities_validated = validate_and_transform(
        entities_raw,
        PoliticalEntityModel,
        config
    )

    print("[1/4] Loading Political Entities...")
    load_to_duckdb("political_entities", entities_validated, config)

    # ------------------------------------------------------------------------
    # PHASE 2: WARS
    # ------------------------------------------------------------------------
    print("\n[2/4] Extracting Wars...")
    wars_raw = extract_cow_wars(config)

    print("[2/4] Validating Wars...")
    wars_validated = validate_and_transform(wars_raw, WarModel, config)

    print("[2/4] Loading Wars...")
    load_to_duckdb("wars", wars_validated, config)

    # ------------------------------------------------------------------------
    # PHASE 3: BELLIGERENT LINKS
    # ------------------------------------------------------------------------
    print("\n[3/4] Extracting Belligerent Links...")
    links_raw = extract_belligerent_links(config)

    print("[3/4] Loading Belligerent Links...")
    # Note: Would need BelligerentLinkModel for full validation
    load_to_duckdb("belligerent_links", links_raw, config)

    # ------------------------------------------------------------------------
    # PHASE 4: DATA QUALITY CHECKS
    # ------------------------------------------------------------------------
    print("\n[4/4] Running Data Quality Checks...")
    run_quality_checks(config)

    print("\n" + "=" * 80)
    print("PIPELINE COMPLETE")
    print("=" * 80)


def run_quality_checks(config: PipelineConfig) -> None:
    """Run data quality validation queries."""
    conn = duckdb.connect(str(config.duckdb_path))

    checks = {
        "Orphaned wars (no belligerents)": """
            SELECT w.war_id, w.name
            FROM wars w
            LEFT JOIN belligerent_links bl ON w.war_id = bl.target_id
            WHERE bl.source_id IS NULL
        """,

        "Entities with impossible dates": """
            SELECT entity_id, name, existence_start, existence_end
            FROM political_entities
            WHERE existence_end < existence_start
        """,

        "Belligerents with dates outside war bounds": """
            SELECT bl.source_id, bl.target_id, bl.entry_date, w.start_date
            FROM belligerent_links bl
            JOIN wars w ON bl.target_id = w.war_id
            WHERE bl.entry_date < w.start_date
               OR bl.exit_date > w.end_date
        """,
```

```
        "Wars with zero casualties": """
            SELECT war_id, name, total_military_deaths_estimated
            FROM wars
            WHERE total_military_deaths_estimated = 0
        """
    }

    for check_name, query in checks.items():
        result = conn.execute(query).fetchall()
        if result:
            print(f"⚠️  {check_name}: {len(result)} issues")
        else:
            print(f"✓  {check_name}: PASS")

    conn.close()


if __name__ == "__main__":
    run_pipeline()
```

3. INCREMENTAL UPDATE STRATEGY

```
# Add to pipeline for ongoing data ingestion

@dataclass
class IncrementalConfig:
    """Configuration for incremental updates."""

    last_update_timestamp: Optional[datetime] = None
    update_mode: str = "UPSERT"  # UPSERT, APPEND, REPLACE
    conflict_resolution: str = "NEWEST_WINS"  # NEWEST_WINS, MANUAL_REVIEW


def incremental_load_ucdp(config: PipelineConfig) -> None:
    """Load latest UCDP data incrementally.

    UCDP updates annually. Check for new data and merge.
    """
    conn = duckdb.connect(str(config.duckdb_path))

    # Get max date from existing wars
    max_date = conn.execute("""
        SELECT MAX(end_date) FROM wars WHERE ucdp_conflict_id IS NOT NULL
    """).fetchone()[0]

    print(f"Last UCDP data: {max_date}")

    # Fetch new UCDP data
    import pandas as pd
    df = pd.read_csv(config.ucdp_url)

    # Filter to new/updated conflicts
    df_new = df[pd.to_datetime(df["year"], format="%Y") > pd.to_datetime(max_date)]

    print(f"Found {len(df_new)} new/updated conflicts")

    # Transform and upsert
    # ... (implementation similar to extract_cow_wars)

    conn.close()
```

4. EXPORT TO FOUNDRY

```
def export_to_foundry_json(config: PipelineConfig) -> None:
    """Export DuckDB to Foundry-compatible JSON.

    Creates separate JSON files for each object type and link type.
    """
    conn = duckdb.connect(str(config.duckdb_path))

    # Export political entities
    entities = conn.execute("SELECT * FROM political_entities").df()
```

```
    entities.to_json(
        config.output_dir / "political_entities.json",
        orient="records",
        date_format="iso"
    )

    # Export wars
    wars = conn.execute("SELECT * FROM wars").df()
    wars.to_json(
        config.output_dir / "wars.json",
        orient="records",
        date_format="iso"
    )

    # Export links
    links = conn.execute("SELECT * FROM belligerent_links").df()
    links.to_json(
        config.output_dir / "belligerent_links.json",
        orient="records",
        date_format="iso"
    )

    print(f"✓ Exported {len(entities)} entities")
    print(f"✓ Exported {len(wars)} wars")
    print(f"✓ Exported {len(links)} links")
    print(f"\nFiles ready for Foundry upload at: {config.output_dir}")

    conn.close()
```

5. CRITICAL GAPS & MANUAL WORK REQUIRED

```
### DATA GAPS REQUIRING SME INPUT

1. **Pre-1816 Entities** (COW cutoff)
   - Manual entry: Roman Empire, Mongol Empire, etc.
   - Source: Wikipedia + Turchin's Seshat database
   - Effort: ~40 hours for major empires

2. **Battle-Level Data** (Scarce for pre-20th century)
   - Clodfelter book digitization: ~200 hours
   - Alternative: Contract Rand/CNA for historical battle DB

3. **Non-State Actors** (ISIS, Wagner, Cartels)
   - No authoritative dataset exists
   - Build from SIGACT + open-source intelligence
   - Effort: ~80 hours per major conflict

4. **Classified Data Integration**
   - SIPRNET SIGACT requires SECRET environment
   - JWICS OPSREPS for ongoing operations
   - Cannot automate - manual exports required

5. **Equipment Losses** (ORYX-style)
   - Open-source: https://www.oryxspioenkop.com/
   - Requires web scraping + SME validation
   - Ukraine data: ~2000 records manually verified

### PRIORITY ORDER

**Week 1**: COW + UCDP pipeline (automated)
**Week 2-3**: Historical battles from Clodfelter (manual)
**Week 4**: Modern conflicts SIGACT integration (clearance required)
**Month 2+**: Non-state actors, equipment losses (ongoing)
```

Bottom Line: You now have a production ontology with a realistic ingestion plan. The automated portion (COW/UCDP) gives you 1816-2023 state-level conflicts in ~4 hours of compute time. Everything else requires either clearance access or manual digitization.

Challenge: Are you building this for historical analysis or live operational planning? If the latter, we need real-time SIGACT streaming, not batch ETL.