

Good — I've thought this through. Before showing the design, here's a quick summary of the analytical logic, followed by the formal ATWL specification.

Natural-language description of the workflow

1. Data preparation. Starting from the paper records, the workflow first extracts the disambiguated **authors** as analysis units (using the DBLP identifiers) and partitions the papers into **overlapping multi-year time windows** (e.g., 5-year sliding windows). For each window, a **weighted co-authorship graph** is built among the authors active in that window (edge weight = number of jointly authored papers).

2. Community detection (with human-in-the-loop tuning). A community-detection algorithm (e.g., Louvain/Leiden) is applied to each window's co-authorship graph to produce candidate communities. The analyst visually inspects each window's network with community colouring, judges whether the granularity and separation are meaningful (e.g., communities correspond to recognisable research groups), and — if not — adjusts the algorithm's resolution parameter and re-runs. This loop continues until the communities are judged interpretable and well separated.

3. Tracking communities over time. Once communities are stable within each window, their **membership overlap** across consecutive windows is computed (e.g., Jaccard similarity), and matched community instances are linked into longitudinal **community lineages** (tracks), which may branch (split) or converge (merge). The analyst inspects these lineages, together with the per-window network visualisations, to abstract and label **lifecycle events**: persistence, merge, split, dissolution, emergence. An alluvial/Sankey visualisation of the tracked communities over time supports this interpretation and directly answers questions about membership overlap and evolution.

4. Core stability and leadership. For each community track, the persistence of each author's membership across windows is computed, distinguishing a **stable core** from peripheral/moving members. Core members are then ranked by productivity and citation influence to identify **leaders**. A timeline/heatmap visualisation of core-member presence lets the analyst judge each community's stability and formulate statements about leadership continuity and turnover.

5. Research foci and their evolution. A topic model is fit on titles/abstracts/keywords across the whole corpus. Topic proportions are aggregated per community instance and window to obtain an evolving **topical profile** for each community track. A streamgraph visualises how each community's topic mix shifts over time; the analyst interactively names

the dominant research foci and assesses whether the labelling is distinct and stable, refining the topic model if needed.

6. Synthesis. Finally, the analyst integrates the community-tracking results, lifecycle events, core-stability findings, and research-focus evolution into a narrative characterising the intellectual structure of the IEEE VIS community: how many communities exist, how stable their cores/leaders are, what they work on, and how/when they merge, split, or dissolve.

ATWL specification

atwl

```
workflow collaboration-community-evolution
  template: define-unit (extract authors) -> define-unit (time-partition
    characterise (co-authorship network) ->
    loop(define-unit (community detection) -> visualise -> asses
      generate-knowledge (parameter refine)) ->
    characterise (community overlap) -> define-unit (track linea
    abstract (lifecycle events) -> contextualise -> visualise (e
    characterise (core & leadership) -> visualise (core stabilit
    generate-knowledge (stability) ->
    build-model (topic model) -> characterise (topic profiles) -
    loop(visualise (foci evolution) -> abstract (name foci) -> a
      build-model (refine topics)) ->
    generate-knowledge (synthesis)
  description: "Identify collaboration communities among IEEE VIS author
    time, track their evolution (persistence, merges, splits, dissolutio
    characterise the stability of their core/leadership, and relate their
    evolving research foci."

# =====
# INPUT
# =====

artifact D_papers : entities
  origin: given
  internal structure: elementary
  embedment: time
  features:
    - id: authors
```

```
    value structure: list
    value type: reference
    description: "DBLP-disambiguated author identifiers"
  - id: title
    value structure: atomic
    value type: text
    description: "Paper title"
  - id: abstract
    value structure: atomic
    value type: text
    description: "Paper abstract"
  - id: keywords
    value structure: list
    value type: text
    description: "Author/venue keywords"
  - id: citations
    value structure: list
    value type: reference
    description: "Identifiers of cited papers"
description: "IEEE VIS papers (1990-2024) with authorship, metadata, a
citation links"
```

```
# =====
# STEP 1: EXTRACT AUTHORS AND TIME WINDOWS
# =====
```

```
transform T_extract_authors :
  intent: define-unit
  manner: "extract"
  input: D_papers
  output: D_authors
  actor: machine
  description: "Extract unique disambiguated authors as analysis units f
    the paper author lists"
```

```
artifact D_authors : entities
  internal structure: elementary
  embedment: set
  features:
    - id: active_years
      value structure: list
      value type: temporal
```

```

        description: "Years in which the author published"
    - id: paper_count
      value structure: atomic
      value type: numeric
      description: "Total number of papers authored"
    - id: total_citations
      value structure: atomic
      value type: numeric
      description: "Aggregate citation count"
  description: "Disambiguated authors extracted from paper records"

artifact S_windowing : specification
  origin: given
  representation form: "parameter settings"
  description: "Time window length and overlap (e.g., 5-year windows adv
    by 1 year) used to construct temporal snapshots of collaboration"

transform T_partition_time :
  intent: define-unit
  manner: "time-partitioning with overlapping windows"
  input: D_papers, S_windowing
  output: D_windows
  actor: hybrid
  description: "Partition papers into overlapping multi-year time window
    serve as temporal snapshots for network construction"

artifact D_windows : entities
  internal structure: group/cluster
  embedment: time
  features:
    - id: window_span
      value structure: atomic
      value type: temporal
      description: "Start and end year of the window"
  description: "Overlapping time windows used as temporal snapshots"

# =====
# STEP 2: BUILD CO-AUTHORSHIP NETWORKS
# =====

transform T_build_networks :
  intent: characterise

```

```

manner: "define-relations (co-authorship)"
input: D_windows, D_authors, D_papers
output: F_coauth_network
actor: machine
description: "For each window, build a weighted co-authorship graph ar
    authors active in that window (edge weight: number of co-authored pa

artifact F_coauth_network : feature(D_windows)
    value structure: relational configuration
    representation form: "weighted co-authorship graph per window"
    description: "Temporal sequence of co-authorship networks, one per win

# =====
# STEP 3: COMMUNITY DETECTION LOOP
# =====

artifact S_community_params : specification
    origin: given
    representation form: "parameter settings"
    description: "Community-detection algorithm choice and resolution para

loop L_community_detection:
    purpose: "Detect coherent, interpretable collaboration communities wit
        each window's network"
    until: "Communities show clear separation/cohesion and are judged by t
        analyst to correspond to meaningful collaboration groups"
    body:

        transform T_detect_communities :
            intent: define-unit
            manner: "cluster-by-similarity (community detection)"
            input: F_coauth_network, S_community_params
            output: D_communities
            actor: machine
            description: "Detect densely collaborating groups of authors withi
                each window's co-authorship network"

        artifact D_communities : entities
            internal structure: group/cluster
            embedment: time
            features:
                - id: members

```

```

        value structure: list
        value type: reference
        description: "Author identifiers belonging to this community
            instance"
    - id: community_size
        value structure: atomic
        value type: numeric
        description: "Number of authors in the community"
    description: "Groups of densely collaborating authors within a giv
        time window"

transform T_vis_communities :
    intent: visualise
    manner: "node-link-diagram"
    input: F_coauth_network, D_communities
    output: V_communities
    actor: machine
    description: "Display each window's network with nodes coloured by
        detected community for visual assessment"

artifact V_communities : visualisation(F_coauth_network, D_communities)
    layout: "force-directed layout per time window"
    form: "coloured nodes and edges"
    encoding: "node colour: community membership; edge thickness:
        co-authorship weight"
    description: "Per-window views for inspecting community structure"

transform T_assess_communities :
    intent: assess
    manner: "evaluate-clustering-quality"
    input: V_communities, D_communities
    output: K_community_quality
    actor: human
    description: "Judge whether communities are well-separated, cohesi
        and correspond to recognisable research groups"

artifact K_community_quality : knowledge(D_communities)
    representation form: "quality judgment"
    description: "Assessment of whether community granularity/separati
        adequate or requires parameter adjustment"

if K_community_quality indicates refinement needed:

```

```

then:
  transform T_refine_params :
    intent: generate-knowledge
    manner: "adjust method parameters"
    input: K_community_quality, S_community_params
    output: S_community_params_r
    actor: human
    description: "Revise resolution parameter/algorithm choice"

  artifact S_community_params_r : specification
    representation form: "parameter settings"
    description: "Updated community-detection parameters"

  assign:
    S_community_params := S_community_params_r
else:
  exit loop L_community_detection

end loop L_community_detection

# =====
# STEP 4: TRACK COMMUNITIES OVER TIME
# =====

transform T_match_communities :
  intent: characterise
  manner: "define-relations (membership overlap)"
  input: D_communities
  output: F_community_overlap
  actor: machine
  description: "Compute membership overlap (e.g., Jaccard similarity) between
    communities detected in consecutive/overlapping windows"

artifact F_community_overlap : feature(D_communities)
  value structure: relational configuration
  representation form: "correspondence graph linking community instances
    across consecutive windows by membership overlap"
  description: "Temporal linkage structure between community instances"

transform T_build_tracks :
  intent: define-unit
  manner: "merge-groups (track lineages over time)"

```

```
input: D_communities, F_community_overlap
output: D_community_tracks
actor: hybrid
description: "Link corresponding community instances into longitudinal
lineages, allowing branching (splits) and convergence (merges)"
```

```
artifact D_community_tracks : entities
  internal structure: formation
  embedment: time
  features:
    - id: track_span
      value structure: atomic
      value type: temporal
      description: "Time span over which the lineage is observed"
  description: "Longitudinal communities represented as lineage graphs of
linked community instances"
```

```
transform T_identify_lifecycle_events :
  intent: abstract
  manner: "perception-and-interpretation"
  input: D_community_tracks, F_community_overlap, V_communities
  output: P_lifecycle_events
  actor: hybrid
  description: "Identify and label structural transitions in lineages:
persistence, merge, split, dissolution, emergence"
```

```
artifact P_lifecycle_events : pattern(D_community_tracks)
  representation form: "typed event annotations (persist/merge/split/
dissolve/emerge) with time points"
  description: "Lifecycle events characterising community evolution"
```

```
transform T_contextualise_evolution :
  intent: contextualise
  manner: "relative-time-alignment"
  input: D_community_tracks, D_windows
  output: A_evolution
  actor: machine
  description: "Arrange community-track instances along the window timeline
for flow-based visualisation"
```

```
artifact A_evolution : arrangement(D_community_tracks)
  context: D_windows
```

```

    principle: "chronological ordering of windows, instances stacked by track
    identity"
    description: "Temporal arrangement supporting alluvial visualisation"

transform T_vis_evolution :
    intent: visualise
    manner: "alluvial/Sankey diagram"
    input: A_evolution, P_lifecycle_events, F_community_overlap
    output: V_evolution
    actor: machine
    description: "Show community membership flows over time, highlighting
    merges, splits, and dissolutions"

artifact V_evolution : visualisation(A_evolution, P_lifecycle_events, F_
    layout: "alluvial layout, time on horizontal axis"
    form: "coloured flow ribbons and blocks"
    encoding: "block width: community size; ribbon width: shared membershi
    colour: track identity; markers: lifecycle events"
    description: "Overview of community evolution for analyst interpretati

# =====
# STEP 5: CORE STABILITY AND LEADERSHIP
# =====

transform T_characterise_core :
    intent: characterise
    manner: "compute-statistics (membership persistence)"
    input: D_community_tracks, D_communities, D_authors
    output: F_core_membership
    actor: machine
    description: "Compute, per track, each author's persistence across win
    distinguishing stable core from peripheral/transient members"

artifact F_core_membership : feature(D_community_tracks)
    value structure: list
    value type: {reference, numeric}
    description: "Per-author persistence scores within each community trac

transform T_characterise_leaders :
    intent: characterise
    manner: "compute-statistics (productivity/citation ranking)"
    input: F_core_membership, D_authors

```

```

output: F_leadership_score
actor: machine
description: "Rank core members by productivity and citation influence
            identify leading researchers per track"

artifact F_leadership_score : feature(D_community_tracks)
    value structure: list
    value type: {reference, numeric}
    description: "Ranked core members and leadership scores per community"

transform T_vis_core_stability :
    intent: visualise
    manner: "coloured-marks (presence heatmap)"
    input: F_core_membership, F_leadership_score, D_windows
    output: V_core_stability
    actor: machine
    description: "Timeline/heatmap of core author presence per track across
                windows, highlighting stable leaders vs. fluctuating members"

artifact V_core_stability : visualisation(F_core_membership, F_leadership_score)
    layout: "author x window grid, grouped by community track"
    form: "coloured heatmap cells"
    encoding: "cell colour: presence and leadership score; rows grouped by community"
    description: "Support for inspecting core stability and leadership"

transform T_generate_stability_knowledge :
    intent: generate-knowledge
    manner: "formulate-statements"
    input: V_core_stability, F_core_membership, F_leadership_score
    output: K_core_stability
    actor: human
    description: "Formulate statements on core stability, leadership continuity,
                and membership turnover per community"

artifact K_core_stability : knowledge(D_community_tracks, D_authors)
    representation form: "statements"
    description: "Insights on core stability, leadership continuity, and
                membership turnover"

# =====
# STEP 6: RESEARCH FOCI AND THEIR EVOLUTION
# =====

```

```
transform T_topic_model :
  intent: build-model
  manner: "fit-topic-model"
  input: D_papers
  output: M_topics
  actor: machine
  description: "Fit a topic model on titles/abstracts/keywords to derive
    latent research topics"

artifact M_topics : model(D_papers)
  model type: "topic model"
  representation form: "topic-term distributions and per-paper topic
    proportions"
  description: "Latent topic structure of the paper corpus"

transform T_characterise_foci :
  intent: characterise
  manner: "aggregate (topic proportions per community-window)"
  input: M_topics, D_community_tracks, D_communities, D_windows
  output: F_topic_profile
  actor: machine
  description: "Aggregate topic proportions of member-authored papers to
    obtain an evolving topical profile per community track"

artifact F_topic_profile : feature(D_community_tracks)
  value structure: matrix
  value type: numeric
  representation form: "community-track x topic x window proportions"
  description: "Evolving topical profile of each community lineage"

loop L_foci_interpretation:
  purpose: "Assign and validate interpretable research-focus labels for
    community's evolving topical profile"
  until: "Labels are judged distinct, stable, and accurately reflective
    each community's evolving foci"
  body:

    transform T_vis_foci_evolution :
      intent: visualise
      manner: "streamgraph"
      input: F_topic_profile, D_windows
```

```

output: V_foci_evolution
actor: machine
description: "Display evolving topical composition of each community
track as a streamgraph across windows"

artifact V_foci_evolution : visualisation(F_topic_profile, D_windows)
layout: "streamgraph, time on horizontal axis"
form: "stacked coloured bands"
encoding: "band thickness: topic proportion; colour: topic identity
faceted by community track"
description: "Support for interpreting shifting research foci"

transform T_name_foci :
intent: abstract
manner: "name-patterns"
input: V_foci_evolution, F_topic_profile, M_topics
output: P_research_foci
actor: human
description: "Assign concise research-focus labels to dominant topics
per community and period"

artifact P_research_foci : pattern(F_topic_profile)
representation form: "labelled sequences of dominant research foci
community track"
description: "Human-interpreted research-focus labels and their
evolution"

transform T_assess_foci :
intent: assess
manner: "judge-visualisation-adequacy"
input: V_foci_evolution, P_research_foci
output: K_foci_quality
actor: human
description: "Judge whether labels are distinct, stable, and accurate"

artifact K_foci_quality : knowledge(P_research_foci)
representation form: "quality judgment"
description: "Assessment of labelling adequacy"

if K_foci_quality indicates refinement needed:
then:
transform T_refine_topics :

```

```

        intent: build-model
        manner: "adjust topic model parameters"
        input: K_foci_quality, D_papers, M_topics
        output: M_topics_r
        actor: hybrid
        description: "Adjust topic count/preprocessing and refit the m

artifact M_topics_r : model(D_papers)
    model type: "topic model"
    representation form: "topic-term distributions and per-paper t
        proportions"
    description: "Refined topic model"

assign:
    M_topics := M_topics_r
else:
    exit loop L_foci_interpretation

end loop L_foci_interpretation

# =====
# STEP 7: SYNTHESISE FINDINGS
# =====

transform T_synthesise :
    intent: generate-knowledge
    manner: "formulate-statements"
    input: D_community_tracks, P_lifecycle_events, K_core_stability,
        P_research_foci, V_evolution
    output: K_findings
    actor: human
    description: "Synthesise the number and stability of communities, lead
        continuity, evolving research foci, and merge/split/dissolution hist

artifact K_findings : knowledge(D_community_tracks, P_research_foci)
    representation form: "statements and narrative summary"
    description: "Integrated account of the IEEE VIS community's collabora
        structure: major communities, stable cores/leaders, evolving research
        foci, and merge/split/dissolution events"

```