

Analytical Workflow: Discovering and Characterising Collaboration Communities in the IEEE VIS Literature

Analytical logic (natural-language overview)

The task has two intertwined layers: (1) detecting **who collaborates with whom, and how those groupings change over a 35-year span**, and (2) **interpreting** those groupings — their stability, leadership, research focus, and their merge/split/dissolve dynamics. This calls for a workflow that alternates between computational network/text analysis and human interpretation, with several rounds of parameter calibration. I model it in six stages:

A. Data preparation. From the paper records, I extract authors as first-class analysis units and partition the publication timeline into time windows (the "unit of temporal resolution" at which communities will be defined). For each window I build a co-authorship network (nodes = active authors, edges = weighted co-authorship). This treats "papers" and "authors" as different levels of analysis, following the convention (used throughout the library, e.g. in *cluster-calendar* and *reducing-snapshots-to-points*) of first defining the right analysis unit before computing anything else.

B. Per-window community detection (iterative). A graph community-detection algorithm (e.g. Louvain) is applied to each window's network. Because resolution parameters strongly affect whether communities are meaningful (too fragmented / too coarse), this is wrapped in a refinement loop with visual assessment of modularity, size distribution and interpretability — directly modelled on the calibration loop in *cluster-calendar* and the parameter-adjustment loops in *MobilityGraphs*.

C. Cross-window lineage tracking (iterative). Communities detected independently in each window are not yet "the same community over time." I compute pairwise membership overlap (Jaccard similarity) between communities in temporally adjacent windows and link them above a calibrated threshold into **lineages** — persistent community identities — while explicitly classifying each linkage as *continuation*, *merge*, *split*, *birth*, or *dissolution*. This is the structural heart of the workflow and is modelled closely on *EventThread*'s tensor-decomposition-into-threads-with-branching/merging pattern (there: stage threads; here: window lineages) and visualised as a column-band diagram in the manner of *StratomeX* (there: overlapping patient stratifications; here: overlapping community memberships across adjacent windows), overlaid with an alluvial/flow view as in *EventThread* and *SOMFlow*. The overlap/matching threshold is itself calibrated in a small loop, mirroring the parameter-refinement pattern in *ST_analysis_modelling*.

D. Core stability and leadership. For each lineage, I compute each member's presence frequency across the lineage's windows to distinguish a **stable core** from transient participants, and compute within-community centrality (degree/betweenness) combined with publication/citation volume to identify **leaders** per window. This is visualised as an author × time-window presence matrix (heatmap), directly analogous to the state-persistence and role-identification steps in *DPVis* and the actor-role analysis in *NodeTrix*, whose hybrid node-link/matrix representation is also recommended for drilling into any single dense community instance.

E. Research foci via topic modelling (iterative). Titles, abstracts and keywords of papers authored by each community's members (within each window) are aggregated into a corpus per community instance, and a topic model (NMF) is fit and iteratively refined — following the interactive topic-modelling loop of *UTOPIAN (interactive-topic-modeling)* — producing named research-focus labels that can be tracked along each lineage's timeline to show topical evolution or drift.

F. Individual-level mobility and cross-community overlap. Beyond window-to-window community overlap (stage C), I reconstruct each author's personal trajectory through lineages over their career (which lineages they belonged to, whether simultaneously or sequentially), revealing bridging authors and multi-community membership — addressing "how does membership overlap change over time" at the individual level.

G. Integrated interpretation loop. The analyst brings the evolution diagram, stability heatmap, topic timelines and mobility view together to interpret *why* particular merges/splits/dissolutions occurred (e.g., a split coinciding with a topical bifurcation, or a dissolution coinciding with core-member attrition), iterating on window size or matching thresholds if the picture is unclear — an "implicit termination" loop, since the stopping criterion ("sufficient holistic understanding") is not reducible to a single assessable artifact, following the convention used for exploratory synthesis loops in e.g. *MotionExplorer* and *episode-based-temporal-analysis*.

H. Knowledge synthesis. Final statements about the number and character of communities, core stability, leadership, foci evolution, overlap patterns, and merge/split/dissolve events with their likely drivers.

ATWL Representation

plaintext

workflow vis-community-evolution-analysis

```
template: define-unit (windows, authors, networks)
  loop(define-unit (detect communities)  characterise (quality
    visualise  assess)
  characterise (overlaps)
  loop(define-unit (match lineages)  contextualise  visualise
    assess)
  characterise (core & leaders)  visualise  abstract
  loop(build-model (topics)  visualise  abstract  assess)
  characterise (author trajectories)  contextualise  visualise
  abstract
  loop(abstract (integrate)  assess)
  generate-knowledge
```

description: "Identify communities of collaborating researchers in the IEEE VIS literature (1990-2024) via time-windowed co-authorship networks, track their evolution (continuation, merger, splitting, birth, dissolution) across windows, characterise core stability, leadership, research foci, and cross-community researcher mobility"

```
# =====
# INPUT
# =====
```

```
artifact D_papers : entities
  origin: given
  internal structure: elementary
  embedment: {set, time}
  features:
    - id: f_authors
      value structure: list
      value type: reference
      description: "DBLP-disambiguated author identifiers"
    - id: f_year
      value structure: atomic
      value type: temporal
      description: "Publication year"
    - id: f_title
      value structure: atomic
      value type: text
      description: "Paper title"
```

- id: f_abstract
 - value structure: atomic
 - value type: text
 - description: "Paper abstract"
- id: f_keywords
 - value structure: list
 - value type: text
 - description: "Author-supplied keywords"
- id: f_citations
 - value structure: atomic
 - value type: numeric
 - description: "Citation count"

description: "3596 IEEE VIS/InfoVis/VAST/SciVis papers (1990-2024) with disambiguated author lists"

artifact S_window_config : specification

- origin: given
- representation form: "parameter settings"
- description: "Initial time-window scheme: window length and stride (e.g. 3-year sliding windows, 1-year stride) for constructing successive co-authorship snapshots"

```
# =====
# PHASE A: UNITS OF ANALYSIS - WINDOWS, AUTHORS, NETWORKS
# =====
```

transform T_define_windows :

- intent: define-unit
- manner: "time-partitioning into sliding windows"
- input: D_papers, S_window_config
- output: D_time_windows
- actor: machine
- description: "Partition the publication timeline into overlapping multi-year windows serving as the temporal resolution for community detection"

artifact D_time_windows : entities

- internal structure: episode
- embedment: time
- features:
 - id: f_window_id
 - value structure: atomic

value type: categorical
description: "Window identifier"
- id: f_year_range
value structure: vector
value type: temporal
description: "Start and end year of the window"
description: "Bounded multi-year intervals over which co-authorship networks and communities are computed"

transform T_extract_authors :
intent: define-unit
manner: "extract"
input: D_papers
output: D_authors
actor: machine
description: "Extract unique disambiguated authors as first-class analysis units"

artifact D_authors : entities
internal structure: elementary
embedment: set
features:
- id: f_total_papers
value structure: atomic
value type: numeric
description: "Total number of papers authored"
- id: f_career_span
value structure: vector
value type: temporal
description: "First and last publication year"
description: "Individual researchers, disambiguated via DBLP identifier"

transform T_build_networks :
intent: define-unit
manner: "aggregate co-authorship into per-window network snapshots"
input: D_papers, D_authors, D_time_windows
output: D_networks
actor: machine
description: "Construct one weighted co-authorship network per time window: nodes are authors active in the window, edges are weighted by number of jointly authored papers"

```

artifact D_networks : entities
  internal structure: formation
  embedment: time
  features:
    - id: f_coauthor_graph
      value structure: relational configuration
      value type: {reference, numeric}
      representation form: "weighted undirected co-authorship graph"
      description: "Author nodes with co-authorship-count edge weights f
        this window"
  description: "Temporal sequence of co-authorship network snapshots, on
    per time window"

# =====
# PHASE B: PER-WINDOW COMMUNITY DETECTION (ITERATIVE CALIBRATION)
# =====

artifact S_community_params : specification
  origin: given
  representation form: "parameter settings"
  description: "Community-detection algorithm choice and resolution
    parameter (e.g. Louvain modularity resolution) applied uniformly acr
    all window networks"

loop L_community_detection:
  purpose: "Calibrate community-detection resolution so that communities
    are neither over-fragmented nor overly coarse, and remain interpreta
    across the whole time span"
  until: "Modularity, size distribution, and visual interpretability of
    communities are satisfactory across the majority of time windows"
  body:

    transform T_detect_communities :
      intent: define-unit
      manner: "graph community detection (modularity optimisation, e.g.
        Louvain), applied independently per window"
      input: D_networks, S_community_params
      output: D_communities, F_community_label
      actor: machine
      description: "Partition each window's co-authorship network into
        communities of densely collaborating authors"

```

```

artifact D_communities : entities
  internal structure: group/cluster
  embedment: {set, time}
  features:
    - id: f_window_id
      value structure: atomic
      value type: reference
      description: "Time window this community instance belongs to"
    - id: f_size
      value structure: atomic
      value type: numeric
      description: "Number of member authors"
  description: "Community instances: groups of authors detected as
    densely collaborating within one specific time window"

artifact F_community_label : feature(D_authors)
  value structure: vector
  value type: categorical
  description: "Per-window community-assignment sequence for each
    author across all time windows in which they were active"

transform T_compute_quality :
  intent: characterise
  manner: "compute-statistics (modularity, size distribution)"
  input: D_communities, D_networks
  output: F_quality
  actor: machine
  description: "Compute per-window modularity score, community count
    mean/variance of community size, and singleton-author fraction"

artifact F_quality : feature(D_networks)
  value structure: vector
  value type: numeric
  description: "Per-window community-detection quality metrics"

transform T_visualise_communities :
  intent: visualise
  manner: "small-multiples node-link / hybrid NodeTrix-style views p
    window"
  input: D_networks, D_communities, F_community_label
  output: V_communities
  actor: machine

```

```
description: "Display each window's network with communities
  colour-coded; densest communities optionally shown as matrices
  (NodeTrix-style) to reveal internal collaboration structure"

artifact V_communities : visualisation(D_networks, D_communities,
  F_community_label)
  layout: "small multiples, one panel per time window"
  form: "node-link diagrams with colour-coded community membership;
    NodeTrix-style hybrid matrix insets for dense communities"
  encoding: "node colour: community label; node size: author
    publication count; matrix cell: intra-community co-authorship
    count"
  description: "Per-window overview supporting visual judgment of
    community coherence and granularity"

transform T_assess_communities :
  intent: assess
  manner: "evaluate-clustering-quality"
  input: V_communities, F_quality
  output: K_community_quality
  actor: human
  description: "Judge whether modularity, size distribution, and
    visual coherence of communities are acceptable across windows"

artifact K_community_quality : knowledge(D_communities)
  representation form: "quality judgment"
  description: "Assessment of whether current resolution parameter
    yields interpretable, appropriately granular communities"

if K_community_quality indicates refinement needed:
  then:
    transform T_adjust_community_params :
      intent: generate-knowledge
      manner: "adjust resolution parameter based on assessment"
      input: K_community_quality, S_community_params
      output: S_community_params_r
      actor: human
      description: "Revise resolution or algorithm choice to correct
        over-fragmentation or excessive coarseness"

    artifact S_community_params_r : specification
      representation form: "parameter settings"
```

```

        description: "Updated community-detection configuration"

    assign:
        S_community_params := S_community_params_r
    else:
        exit loop L_community_detection

end loop L_community_detection

# =====
# PHASE C: CROSS-WINDOW LINEAGE TRACKING (ITERATIVE CALIBRATION)
# =====

transform T_compute_overlaps :
    intent: characterise
    manner: "define-relations (similarity) – Jaccard overlap of member set"
    input: D_communities
    output: F_overlap
    actor: machine
    description: "Compute pairwise membership-overlap similarity between
        every pair of community instances in temporally adjacent windows"

artifact F_overlap : feature(D_communities)
    value structure: relational configuration
    value type: numeric
    representation form: "directed overlap graph (Jaccard similarity) between
        community instances of temporally adjacent windows"
    description: "Basis for linking community instances into persistent
        lineages"

artifact S_matching_params : specification
    origin: given
    representation form: "parameter settings"
    description: "Overlap threshold above which two community instances in
        adjacent windows are linked as continuous, and rules for classifying
        one-to-many / many-to-one links as split / merge"

loop L_lineage_tracking:
    purpose: "Calibrate the overlap threshold so that resulting community
        lineages and their merge/split/birth/dissolution events are
        structurally plausible"
    until: "The evolution diagram shows neither excessive fragmentation

```

(spurious births/dissolutions) nor implausible over-merging"
body:

```
transform T_match_lineages :  
  intent: define-unit  
  manner: "link community instances above overlap threshold into  
    lineages; classify each linkage as continuation, merge, split,  
    birth, or dissolution"  
  input: D_communities, F_overlap, S_matching_params  
  output: D_lineages, F_transition_events  
  actor: machine  
  description: "Track community identity across windows, producing  
    persistent lineages and typed transition events between  
    consecutive windows"  
  
artifact D_lineages : entities  
  internal structure: sequence  
  embedment: set  
  features:  
    - id: f_first_window  
      value structure: atomic  
      value type: reference  
      description: "Window in which the lineage first appears"  
    - id: f_last_window  
      value structure: atomic  
      value type: reference  
      description: "Window in which the lineage last appears (or  
        dissolves)"  
  description: "Ordered sequences of community instances across time  
    windows, each sequence representing one persistent community"  
  
artifact F_transition_events : feature(D_lineages)  
  value structure: vector  
  value type: categorical  
  description: "Per-consecutive-window-pair transition type along ea  
    lineage: continuation, merge, split, birth, or dissolution"  
  
transform T_arrange_evolution :  
  intent: contextualise  
  manner: "column-band arrangement (StratomeX-style) plus alluvial  
    flow layout (EventThread-style)"  
  input: D_lineages, D_communities, F_overlap, F_transition_events
```

```
output: A_evolution
actor: machine
description: "Arrange community instances as columns per time window
with connecting bands sized by membership overlap, and as an
alluvial diagram highlighting branch/merge/dissolution points"

artifact A_evolution : arrangement(D_lineages)
context: D_time_windows
principle: "horizontal: time windows as columns; vertical: stacked
community-instance blocks sized by member count; bands between
adjacent columns sized by overlap; alluvial flow shows lineage
continuity"
description: "Temporal arrangement exposing continuity, merges,
splits, births, and dissolutions of communities"

transform T_visualise_evolution :
intent: visualise
manner: "column-band matrix + alluvial diagram"
input: A_evolution
output: V_evolution
actor: machine
description: "Render the community-evolution diagram with
colour-coded lineages and highlighted transition events"

artifact V_evolution : visualisation(A_evolution)
layout: "sequence of window columns with connecting overlap bands;
alluvial flow overlay"
form: "stacked blocks, ribbon bands, branch/merge glyphs"
encoding: "block height: community size; band width: overlap
strength; block colour: lineage identity; glyph: transition type
(merge/split/birth/dissolution)"
description: "Primary view for inspecting how communities persist,
combine, fragment, or disappear over the 35-year span"

transform T_assess_lineages :
intent: assess
manner: "evaluate structural plausibility of lineages and events"
input: V_evolution, F_transition_events
output: K_lineage_quality
actor: human
description: "Judge whether the overlap threshold produces
believable lineages, avoiding spurious fragmentation or
```

```

        over-merging"

artifact K_lineage_quality : knowledge(D_lineages)
    representation form: "quality judgment"
    description: "Assessment of whether current matching threshold
        yields plausible lineage structure and transition events"

if K_lineage_quality indicates refinement needed:
    then:
        transform T_adjust_matching_params :
            intent: generate-knowledge
            manner: "adjust overlap threshold based on assessment"
            input: K_lineage_quality, S_matching_params
            output: S_matching_params_r
            actor: human
            description: "Revise the overlap threshold to correct
                fragmentation or over-merging"

        artifact S_matching_params_r : specification
            representation form: "parameter settings"
            description: "Updated lineage-matching configuration"

        assign:
            S_matching_params := S_matching_params_r
    else:
        exit loop L_lineage_tracking

end loop L_lineage_tracking

# =====
# PHASE D: CORE STABILITY AND LEADERSHIP
# =====

transform T_compute_core :
    intent: characterise
    manner: "compute-statistics (per-author presence frequency; centrality
    input: D_lineages, D_communities, D_networks, F_community_label
    output: F_core_members, F_leaders
    actor: machine
    description: "For each lineage, compute each author's presence frequen
        across the lineage's windows to identify a stable core; compute
        within-community degree/betweenness centrality combined with

```

publication and citation counts to identify leading researchers per window"

artifact F_core_members : feature(D_lineages)

value structure: list

value type: reference

description: "Authors present in a majority of the lineage's time windows, forming its stable core"

artifact F_leaders : feature(D_communities)

value structure: list

value type: reference

description: "Most central / most cited authors within each community instance"

transform T_visualise_stability :

intent: visualise

manner: "author x time-window presence heatmap per lineage"

input: F_core_members, F_leaders, D_lineages

output: V_stability

actor: machine

description: "Display, per lineage, which authors were present in which windows, with core members and leaders highlighted"

artifact V_stability : visualisation(F_core_members, F_leaders, D_lineages)

layout: "matrix per lineage: rows = authors, columns = time windows"

form: "coloured cells with leader markers"

encoding: "cell colour: membership presence; marker: leader status;

row ordering: total presence count (core members at top)"

description: "Reveals which lineages have a persistent core of leading researchers versus high membership turnover"

transform T_abstract_stability :

intent: abstract

manner: "perception-and-interpretation"

input: V_stability, F_transition_events

output: P_stability_patterns

actor: human

description: "Identify lineages with strong stable cores versus high turnover, and relate core/leader changes to transition events"

artifact P_stability_patterns : pattern(D_lineages)

```

representation form: "descriptions of stability and leadership pattern
description: "Characterisation of each lineage's core stability and
    leadership continuity or change"

# =====
# PHASE E: RESEARCH FOCI VIA TOPIC MODELLING (ITERATIVE)
# =====

transform T_aggregate_text :
  intent: characterise
  manner: "aggregate"
  input: D_communities, D_papers
  output: F_text_corpus
  actor: machine
  description: "Aggregate titles, abstracts, and keywords of papers
    authored by each community instance's members within its window"

artifact F_text_corpus : feature(D_communities)
  value structure: atomic
  value type: text
  description: "Concatenated title/abstract/keyword text for each commun
    instance's papers"

artifact S_topic_params : specification
  origin: given
  representation form: "parameter settings"
  description: "Initial topic-modelling configuration (e.g. number of NM
    topics, term weighting)"

loop L_topic_modeling:
  purpose: "Fit and refine a topic model producing interpretable,
    distinctive research-focus labels for community instances"
  until: "Topics are coherent, distinctive, and interpretable as named
    research foci across community instances"
  body:

    transform T_fit_topics :
      intent: build-model
      manner: "fit-topic-model (NMF)"
      input: F_text_corpus, S_topic_params
      output: M_topic_model, F_topic_weights
      actor: machine

```

```

description: "Fit a topic model over community-instance corpora and
compute each instance's weight distribution across topics"

artifact M_topic_model : model(F_text_corpus)
  model type: "topic model"
  representation form: "term-topic and community-topic weight
    matrices"
  description: "Latent research-theme structure underlying community
    instances' publications"

artifact F_topic_weights : feature(D_communities)
  value structure: vector
  value type: numeric
  description: "Per-community-instance topic weight distribution"

transform T_visualise_topics :
  intent: visualise
  manner: "term-topic matrix + per-lineage topic-weight timeline"
  input: M_topic_model, F_topic_weights, D_lineages
  output: V_topics
  actor: machine
  description: "Display characteristic terms per topic and each
    lineage's topic-weight evolution over its lifespan"

artifact V_topics : visualisation(M_topic_model, F_topic_weights)
  layout: "term-topic matrix; per-lineage stacked/line chart over time"
  form: "coloured cells (term weight); coloured area/line chart"
  encoding: "cell colour: term-topic weight; line/area colour: topic
    identity; height: topic weight within the lineage's window"
  description: "Reveals dominant research foci per community instance
    and how a lineage's focus shifts over time"

transform T_abstract_foci :
  intent: abstract
  manner: "name-patterns"
  input: V_topics
  output: P_research_foci
  actor: human
  description: "Assign interpretable research-focus labels to topics
    (e.g. 'text/NLP visualization', 'immersive analytics')"

artifact P_research_foci : pattern(D_communities, M_topic_model)

```

```

representation form: "named research-focus labels per topic and pe
community instance/lineage"
description: "Human-interpretable characterisation of each
community's dominant research themes and their evolution"

transform T_assess_topics :
  intent: assess
  manner: "evaluate topic coherence and interpretability"
  input: V_topics, P_research_foci
  output: K_topic_quality
  actor: human
  description: "Judge whether topics are distinct and interpretable
as research foci"

artifact K_topic_quality : knowledge(M_topic_model)
  representation form: "quality judgment"
  description: "Assessment of topic coherence, distinctiveness, and
interpretability"

if K_topic_quality indicates refinement needed:
  then:
    transform T_adjust_topic_params :
      intent: generate-knowledge
      manner: "adjust number of topics or term weighting"
      input: K_topic_quality, S_topic_params
      output: S_topic_params_r
      actor: human
      description: "Revise topic-model configuration"

    artifact S_topic_params_r : specification
      representation form: "parameter settings"
      description: "Updated topic-modelling configuration"

    assign:
      S_topic_params := S_topic_params_r
  else:
    exit loop L_topic_modeling

end loop L_topic_modeling

# =====
# PHASE F: INDIVIDUAL-LEVEL MOBILITY AND CROSS-COMMUNITY OVERLAP

```

```
# =====

transform T_derive_trajectories :
  intent: characterise
  manner: "reconstruct per-author lineage-membership sequences"
  input: D_authors, F_community_label, D_lineages
  output: F_author_trajectories
  actor: machine
  description: "Reconstruct each author's career-long sequence of lineage
    memberships, capturing sequential moves between communities and
    simultaneous membership in more than one"

artifact F_author_trajectories : feature(D_authors)
  value structure: vector
  value type: {reference, temporal}
  description: "Per-author ordered sequence of (time window, lineage)
    membership pairs across their career"

transform T_arrange_trajectories :
  intent: contextualise
  manner: "relative-time-alignment against lineage timelines"
  input: F_author_trajectories, D_lineages
  output: A_trajectories
  actor: machine
  description: "Position each author's trajectory against the shared
    lineage timeline for cross-community mobility comparison"

artifact A_trajectories : arrangement(D_authors)
  context: D_lineages
  principle: "horizontal: time; vertical: lineage identity; author
    trajectory drawn as a path across lineage rows"
  description: "Arrangement exposing which authors move between, or
    simultaneously belong to, multiple communities"

transform T_visualise_trajectories :
  intent: visualise
  manner: "trajectory lines over lineage timeline"
  input: A_trajectories
  output: V_trajectories
  actor: machine
  description: "Display author trajectories as paths across the lineage
    timeline, highlighting bridging and multi-community authors"
```

```
artifact V_trajectories : visualisation(A_trajectories)
  layout: "lineage rows along vertical axis; time along horizontal axis"
  form: "coloured trajectory lines/paths, one per selected author or
    author group"
  encoding: "line colour: author identity or role (bridge/core/peripheral)
    vertical jumps: community changes"
  description: "Reveals which researchers bridge multiple communities and
    how cross-community membership evolves"
```

```
transform T_abstract_mobility :
  intent: abstract
  manner: "perception-and-interpretation"
  input: V_trajectories
  output: P_mobility_patterns
  actor: human
  description: "Identify bridging authors, simultaneous multi-community
    membership, and directional flows of researchers between communities"
```

```
artifact P_mobility_patterns : pattern(D_authors, D_lineages)
  representation form: "descriptions of mobility and overlap patterns"
  description: "Characterisation of how individual researchers move
    between, or belong simultaneously to, multiple communities"
```

```
# =====
# PHASE G: INTEGRATED INTERPRETATION
# =====
```

```
loop L_interpretation:
  purpose: "Integrate evolution, stability, research-focus, and mobility
    views to interpret why communities merge, split, or dissolve, revisiting
    earlier configuration choices if the picture remains unclear"
  until: "The analyst has developed a sufficient holistic understanding
    of community evolution and its drivers"
  body:
```

```
    transform T_explore_integrated :
      intent: abstract
      manner: "cross-view comparison"
      input: V_evolution, V_stability, V_topics, V_trajectories
      output: P_integrated_findings
      actor: human
```

```

description: "Interpret merge/split/dissolution events in light of
  co-occurring research-focus shifts, core/leader turnover, and
  researcher mobility"

artifact P_integrated_findings : pattern(D_lineages)
  representation form: "causal interpretations of transition events"
  description: "Explanations linking specific merge/split/dissolutio
    events to concurrent topical shifts, leadership changes, or
    researcher migration"

transform T_assess_completeness :
  intent: assess
  manner: "judge sufficiency of holistic understanding"
  input: P_integrated_findings
  output: K_completeness
  actor: human
  description: "Judge whether all major transition events have been
    satisfactorily interpreted"

artifact K_completeness : knowledge(P_integrated_findings)
  representation form: "quality judgment"
  description: "Assessment of interpretive completeness"

if K_completeness indicates further investigation needed:
  then:
    transform T_refine_focus :
      intent: generate-knowledge
      manner: "decide which earlier configuration to revisit (window
        scheme, matching threshold, or topic granularity)"
      input: K_completeness, S_window_config, S_matching_params
      output: S_window_config_r
      actor: human
      description: "Formulate a revised window or matching
        configuration to resolve ambiguous transitions"

    artifact S_window_config_r : specification
      representation form: "parameter settings"
      description: "Revised window scheme guiding a further pass
        through Phases A-G"

    assign:
      S_window_config := S_window_config_r

```

```

        else:
            exit loop L_interpretation

end loop L_interpretation

# =====
# PHASE H: KNOWLEDGE SYNTHESIS
# =====

transform T_synthesise :
    intent: generate-knowledge
    manner: "formulate-statements"
    input: P_integrated_findings, P_mobility_patterns, P_research_foci,
        P_stability_patterns, F_transition_events
    output: K_findings
    actor: human
    description: "Synthesise findings on the number and character of
        communities, their core stability and leadership, research-focus
        evolution, cross-community overlap, and the timing and drivers of
        merges, splits, and dissolutions"

artifact K_findings : knowledge(P_integrated_findings, P_research_foci)
    representation form: "statements and explanations"
    description: "Documented understanding of the intellectual structure of
        the IEEE VIS community: number of distinct communities, degree of co
        stability and leadership continuity per community, characteristic an
        evolving research foci, patterns of membership overlap and research
        mobility across communities, and identified merge/split/dissolution
        events with plausible drivers"

```