ChatGPT

Opus Core Framework

Authors: Noel Michael Conrad; OpenAI [1]
Date: 27 Feb. 2026
Format: MLA (text-only diagrams; citation-verified; engineering-dense)

Executive Summary

The Opus Core Framework (OCF) is specified here as a **single, authoritative, lossless "system-of-record" architecture description** intended to replace a proliferating internal document ecosystem with a governed, traceable, AI-trainable core. The OCF is designed to be human-readable and machine-actionable: every core statement is typed, uniquely identified, and linked bidirectionally to its provenance and downstream uses, aligning with architecture description conventions (views/viewpoints, stakeholder concerns, required content) and with life-cycle information-item governance. [2]

The consolidation method is grounded in: (a) **architecture description requirements** and viewpoint/concern organization; [3] (b) **life-cycle process and information item discipline**; [4] and (c) **requirements/traceability + configuration/change control** as operational anti-drift mechanisms. [5]

The paper specifies a defensible, reproducible pipeline that—given a corpus—computes near-duplicate clusters (shingling/minhash + containment/resemblance), topic overlap (LDA), redundancy-minimizing extract selection (MMR), salience extraction (TextRank), and document integration centrality (PageRank-style graph ranking). [6] Validation is defined as a combination of (1) **trace coverage** (losslessness) and (2) summary overlap metrics (ROUGE) for derived views, plus stability checks on governance workflows. [7]

Because the Opus folder contents are not provided here, the deliverable is a **methods-and-templates "white paper core"**: schema, thresholds, matrix rules, and governance that can be applied to the real corpus without altering the underlying engineering logic.

Scope and Assumptions

This paper assumes the analyst **does not have direct access to the Opus folder contents** within this drafting environment; therefore, the document provides **methods, thresholds, and templates** rather than asserting corpus-specific outcomes. The consolidation process is defined so it can be executed by a human team, a scripted pipeline, or a hybrid. (Assumption is a design constraint, not a methodological limitation.)

"Lossless" consolidation is defined operationally as: for every source document statement classified as in-scope, there exists (a) a mapped OCF statement (trace-back) and (b) a mapped set of dependent artifacts/uses (trace-forward), or an explicit deprecation decision with rationale that is itself traceable and governed by change control. This mirrors bidirectional traceability expectations and change governance practices for baseline-controlled artifacts. [8]

Citation verification policy in this paper prefers **DOIs and official PDFs/publisher records** where available (e.g., National Aeronautics and Space Administration [9] handbook PDF pages for traceability/ configuration, ACL Anthology PDFs for summarization methods, publisher-indexed DOIs for foundational algorithms). [10]

Standards-Aligned Systems Engineering Foundation

The OCF must be treated as an **architecture description + life-cycle information item set** rather than a "long narrative doc." [11]

Architecture description alignment. ISO/IEC/IEEE 42010:2011 Systems and software engineering—Architecture description [12] specifies a conceptual model of architecture description and requires that architecture descriptions be organized around stakeholder concerns and viewpoints, with defined required contents. [3] In OCF terms: the core is a controlled set of "views" over a single authoritative body of typed statements; viewpoints define conventions for what each view must contain and why. [13]

Life-cycle process alignment. ISO/IEC/IEEE 15288:2015 Systems and software engineering—System life cycle processes [14] establishes a common process framework and terminology for engineering across the system life cycle and supports tailoring and hierarchy. [15] OCF governance (change control, baselines, evidence) is explicitly tied to life-cycle thinking: the OCF is not static documentation; it is a maintained information product that must stay consistent with process execution. [16]

Information item (documentation) alignment. ISO/IEC/IEEE 15289:2019 Systems and software engineering—Content of life-cycle information items [17] specifies the **purpose and content** of life-cycle information items, organized by generic document types and purpose. [18] The OCF schema in this paper implements that intent: each module declares its purpose/type (definition, requirement, interface, rationale, procedure, evidence hook), avoiding untyped prose that cannot be governed or validated. [19]

NASA traceability and change-control grounding. The NASA Systems Engineering Handbook (NASA/ SP-2016-6105 Rev2) [20] explicitly describes requirements management activities that include maintaining **bidirectional traceability** and managing requirement changes via an organization such as a change board, plus maintaining consistency between requirements, ConOps, and architecture/design. [21] It also positions configuration management as a key practice that controls baseline changes and tracks versions, emphasizing consistency between the product and information about the product. [22] It provides a typical change control process (including a configuration control board workflow), which the OCF governance section reuses for documentation baselines. [23]

Opus Core Framework Architecture and Schema

OCF definition

OCF is defined as: a **single authoritative knowledge architecture** composed of typed, uniquely identified statements that can generate consistent "views" (summaries, specs, training prompts, interface descriptions) while remaining governed by baseline and traceability rules. This definition operationalizes the viewpoint/concern requirements of architecture descriptions and the purpose/content expectations for life-cycle information items. [11]

Invariant primitives as the interoperability basis

OCF consolidates prior Opus theoretical domains into six invariant primitives (the “primitive stack”). These are defined as engineering primitives because they can be formalized, parameterized, and tested.

Primitive stack (canonical): - Resonance
- Feedback
- Temporal Phase
- Fractal Recursion
- Agency under Constraint
- Narrative Encoding

Each primitive is treated as a *lens* that can be used to map a module’s claims into variable/constraint form, enabling cross-domain interoperability (human interaction modeling, internal development protocols, AI training grammars). The stack structure serves as a viewpoint system: a “Resonance View,” “Feedback View,” etc., can be generated without duplicating definitions. [13]

OCF module schema

Every element of OCF is a **Module** (the atomic, governable unit). A Module is the smallest unit that can be independently versioned, traced, tested, and deprecated.

Required module fields (minimum): - Module-ID (stable, non-semantic identifier; example format: `OCF-MOD-####`) - Title (human-readable label) - Type (controlled vocabulary): `definition | requirement | constraint | interface | rationale | procedure | evidence-hook | derived-view` - Primitive mapping (one or more of the six primitives) - Viewpoint(s) supported (functional, information, interface, verification, operational, governance) - Statement set (normative “shall/must” and informative “should/may” separated) - Parameters (symbols, units, ranges; defaults and constraints) - Dependencies (Module-IDs) - Provenance map (Source-ID list + location) - Trace-forward map (downstream modules, tests, prompts, implementations) - Verification intent (test/analysis/inspection/demonstration hook) - Status (`draft | baselined | deprecated | superseded`) - Change log entry (who/when/why; linked to change request)

This schema is consistent with the need for disciplined documentation content (typed by purpose) and supports architecture “view” generation by filtering modules by viewpoint and primitive. [24]

Traceability rules

OCF traceability rules are normative:

1. **Trace-back rule:** Every normative statement must trace to at least one Source-ID in a baselined document or be explicitly marked self-derived with documented concurrence/rationale. [25]
2. **Trace-forward rule:** Every requirement/constraint must trace forward to at least one verification intent (test/analysis/inspection/demonstration) and one downstream usage (implementation module, training spec, interface definition, or operational procedure). [25]

3. **No orphan requirements rule:** If a requirement has no acceptable parent/source and is not acceptable as self-derived, it is treated as traceability failure or “gold plating” and must be corrected or removed. [25]
4. **Baseline control rule:** Any change to baselined modules must proceed through configuration change control, including review, disposition, implementation, and documentation release steps. [26]
5. **Duplication resolution rule:** If two modules have equivalent intent, one becomes canonical and the other becomes a derived view or is deprecated; duplication across levels is explicitly resolved. [25]

Text-only architecture diagrams

OCF conceptual stack (ASCII):

```
OCF (authoritative core)
|
|-- Primitive Stack (canonical lenses)
|     1 Resonance
|     2 Feedback
|     3 Temporal Phase
|     4 Fractal Recursion
|     5 Agency under Constraint
|     6 Narrative Encoding
|
|-- Viewpoint Filters (42010-aligned)
|     Functional | Information | Interface | Verification | Operational |
Governance
|
`-- Module Registry (typed, traceable units)
      Definitions | Requirements | Constraints | Interfaces | Rationales |
Procedures | Evidence Hooks | Derived Views
```

Module–provenance–verification relationship (mermaid):

```
classDiagram
  class Module {
    +ModuleID
    +Type
    +PrimitiveMapping
    +Statements
    +Parameters
    +Status
  }

  class SourceDoc {
    +SourceID
```

```
  +Title
  +Version
  +Location
}

class TraceLink {
  +LinkType  // trace-back | trace-forward
  +Strength  // exact | partial | inferred
}

class VerificationArtifact {
  +ArtifactID
  +Method  // test | analysis | inspection | demonstration
  +AcceptanceCriteria
}

Module "1" --> "many" TraceLink : has
TraceLink "many" --> "1" SourceDoc : pointsTo
Module "1" --> "many" VerificationArtifact : verifiedBy
```

Viewpoint coverage map (ASCII template):

```
VIEWPOINT COVERAGE MAP (template)

Functional View:      OCF-MOD-____, OCF-MOD-____, ...
Information View:     OCF-MOD-____, OCF-MOD-____, ...
Interface View:       OCF-MOD-____, OCF-MOD-____, ...
Verification View:    OCF-MOD-____, OCF-MOD-____, ...
Operational View:     OCF-MOD-____, OCF-MOD-____, ...
Governance View:      OCF-MOD-____, OCF-MOD-____, ...
```

Consolidation and De-Redundancy Algorithms

This section specifies the reproducible algorithms and default parameters used to (a) cluster redundancy, (b) select canonical content, and (c) produce defensible deprecation recommendations. Algorithms are chosen for peer-reviewed provenance, operational scalability, and interpretability.

Corpus preprocessing

A standard preprocessing stage is required for all text-based methods:

- Normalize encoding, remove boilerplate (headers/footers), preserve section structure.
- Tokenization: word-level for shingling; sentence segmentation for extractive selection.
- Two representations per document:

- **Lexical** (shingles/minhash/TF-IDF) for near-duplicate detection. [27]
- **Semantic** (sentence embeddings or topic distributions) for thematic overlap and selection diversity. [28]

Near-duplicate detection

Core reference method: resemblance and containment via shingling + sketching, enabling clustering of "roughly the same" documents and "roughly contained" documents. [29]

Recommended defaults (practical parameters): - Shingle type: word shingles - Shingle size: `k = 10` words (default)
- `k = 5` if documents are short (<2 pages) or highly paraphrased - Sketch method: MinHash signatures (or equivalent sketch) - Signature size: `m = 200` hash values (default; increase to 400 for tighter confidence on large corpora) - Similarity metric: estimated Jaccard similarity (`J`) - Thresholds: - **Near-duplicate:** `J ≥ 0.85` → same cluster candidate - **Strong near-duplicate:** `J ≥ 0.92` → collapse unless provenance requires retention - **Containment:** `Containment(A ⊂ B) ≥ 0.90` → treat A as "subset doc" of B and consider deprecation of A

These thresholds are directly motivated by the resemblance/containment framework and typical near-duplicate detection practice. [29]

Scalability option: Locality-sensitive hashing / approximate nearest neighbor to avoid $O(n^2)$ comparisons at large n. [30]

Topic modeling for thematic clustering

Core reference method: Latent Dirichlet Allocation (LDA) to represent each document as a mixture over topics, enabling thematic clusters even when documents are not near-duplicates. [31]

Recommended defaults: - Number of topics: Start grid `K ∈ {20, 30, 50, 80}`; choose minimal K that yields stable interpretability and reduces topic entropy across runs. (LDA instability is known; model selection must be validated.) [32]
- Priors: symmetric Dirichlet with small α for sparse topic mixtures (start α=50/K), β=0.01 (classic defaults; tune to corpus). - Topic-overlap threshold:
- Let θ(d) be topic distribution for doc d
- Use cosine similarity of θ vectors; cluster when `cos(θ(d1), θ(d2)) ≥ 0.80` (default)

Redundancy-minimizing selection for consolidation

Core reference method: Maximal Marginal Relevance (MMR) for selecting content that is relevant while minimizing redundancy relative to already-selected content. [33]

MMR selection rule (conceptual): Select item `Di` maximizing:
`MMR(Di) = λ * Sim(Di, Query/CoreNeed) - (1 - λ) * max_{Dj in Selected} Sim(Di, Dj)`

Recommended defaults: - Unit of selection: sentence or paragraph (prefer paragraph for technical claims) - Similarity: cosine similarity over embeddings or TF-IDF vectors - λ (relevance vs. novelty):

- `λ = 0.65` for consolidation into a canonical module (favor relevance)
- `λ = 0.50` for derived summaries (balance)

Salience extraction for candidate core statements

Core reference method: TextRank graph-based ranking for keyword/sentence extraction. [34]

Recommended defaults: - Build graph on sentences: edge weight = similarity (cosine TF-IDF or embedding similarity) - Damping factor: `d = 0.85` (PageRank-standard; TextRank uses analogous value) [35] - Convergence: `ε = 1e-5` or max iterations 100 - Extract top N sentences per section, then apply MMR to remove redundancy

Integration scoring via graph centrality

To identify "structurally important" documents (those referenced or depended upon by others), construct a document graph and apply PageRank-style centrality.

Core reference method: PageRank-style link analysis described in large-scale web search. [36]

Graph construction (recommended): - Nodes: documents (or modules once OCF exists) - Directed edges: - explicit citations/reference links - shared unique term identifiers (glossary tokens or requirement IDs) - shared interfaces/constraints

PageRank defaults: - Damping factor: `α = 0.85` [37] - Teleportation: uniform over documents in corpus (or restricted to baselined set) - Output: centrality score `PR(doc)`

Interpretation for deprecation: - High PR + unique content → retain or canonicalize - Low PR + high redundancy → archive/delete candidate

Validation metrics

Losslessness is validated by trace coverage, not by ROUGE. The NASA handbook emphasizes bidirectional traceability and baseline change management; these are the primary acceptance controls for "no loss" consolidation. [38]

Two complementary validation layers are recommended:

1. **Trace Coverage Metric (TCM):**
2. For each source doc statement in-scope, mark mapped/unmapped
3. `TCM = mapped_statements / total_in_scope_statements`

4. Target: `TCM ≥ 0.98` before any deletions; remaining 2% must be explicitly justified (out-of-scope, duplicated, or deprecated by governance rule)

5. **ROUGE for derived summaries (secondary):** ROUGE provides overlap-based measures for summary evaluation and is appropriate for testing derived "views," not for proving losslessness. [39]

Relevance and Deprecation Decision System

Rigorous relevance criteria

A document is relevant to OCF if and only if it satisfies at least one of the following **system functions**:

- Defines a canonical term, model variable, unit, or constraint required by one or more modules.
- Specifies an interface, dependency, or verification intent required for downstream implementation or training.
- Contains unique evidence required to justify or validate an OCF claim (e.g., experimental results, formal proofs).
- Represents a baselined decision record whose rationale affects constraints or architecture.

These criteria are aligned with the “purpose and content” discipline for life-cycle information items and with the requirement to maintain consistency and traceability across requirements, architecture, and test artifacts. [40]

Deprecation categories

Deprecation is not a single bucket. OCF uses three controlled actions:

- **Retain (Canonical):** Document remains authoritative or becomes an OCF module set.
- **Archive (Non-authoritative):** Document preserved for history but not used as a source of truth.
- **Delete (Safe removal):** Content fully mapped to OCF with trace coverage and no dependency risk.

These actions rely on baseline/change control discipline to prevent accidental loss through uncontrolled “cleanup.” [41]

Defensible deprecation matrix method

For each document `d`, compute a vector of scores:

- `Rdup(d)` redundancy: max near-duplicate similarity to any other doc (Jaccard estimate)
- `Ccont(d)` containment: max containment in another doc
- `Tsim(d)` topic similarity: max cosine similarity in topic simplex
- `PR(d)` integration centrality: PageRank score in doc graph
- `U(d)` uniqueness: 1 − max semantic redundancy after MMR selection
- `V(d)` verification relevance: binary/score whether doc contains unique verification evidence hooks
- `D(d)` dependency risk: binary/score if name/paths are referenced by code/procedures

Default rule set (deterministic): - If `D(d)=1` or `V(d)=1` → **retain or archive** (never delete without migration) - Else if `Rdup(d)≥0.92` or `Ccont(d)≥0.90` and trace-mapped → **archive** (delete after waiting period) - Else if `PR(d)` is bottom decile and `U(d)` low → **archive** - Else → **retain** until mapped or elevated into OCF modules

These rules instantiate the idea that duplication and low integration predict deprecability, while traceability and dependency risk dominate safety decisions. [42]

Sample Document Deprecation Matrix (hypothetical filenames)

Note: Filenames are illustrative; numeric values are plausible defaults. Replace with computed results.

Hypothetical file	Primitive mapping	PR (0–1)	Redundancy cluster	Max Jaccard	Action	Ratio
`Hyn_System_Treatise.md`	Fractal Recursion	0.18	`CL-04`	0.90	Archive	High conte expre OCF F modu as leg narra ration
`Whole_Holy_Body_Brain.pdf`	Feedback; Fractal Recursion	0.22	`CL-03`	0.88	Archive	Core retain embo feedb const legacy meta remo histor
`Dimensional_Aesthetics_I.md`	Resonance; Fractal Recursion	0.31	`CL-02`	0.86	Retain→Canonicalize	Conta paran for st reson migra interf const
`Dimensional_Aesthetics_II.md`	Resonance; Fractal Recursion	0.12	`CL-02`	0.93	Archive	Near- of DA only i exam porte
`Physiological_Rapport.md`	Resonance; Feedback	0.27	`CL-05`	0.89	Retain→Merge	Merg “Hum Synch mode citatio evide

Hypothetical file	Primitive mapping	PR (0–1)	Redundancy cluster	Max Jaccard	Action	Ratio
`Cross_Linguistic_Rapport.md`	Resonance; Narrative Encoding	0.11	`CL-05`	0.91	Archive	Conta unifie mode after cover
`Heliometric_Modeling.md`	Temporal Phase	0.09	`CL-06`	0.87	Archive	Mode into C tempo functi only u const proce
`Adaptive_Consumer_Narratives.md`	Temporal Phase; Agency	0.14	`CL-06`	0.84	Retain→Refactor	Refac OCF b phase const retair conta datas
`Honor_Valor_Nobility.md`	Agency under Constraint	0.05	`CL-01`	0.62	Archive	Recas const coeffi gover policy interp backg

Implementation Protocol, Governance, and Schedule

Reproducible end-to-end implementation plan

This plan is designed to be executable as a checklist and script spec.

1. **Inventory and normalization**
2. Enumerate files, versions, timestamps, owners
3. Normalize to plain text (retain section structure and headings)
4. Generate Source-IDs: `SRC-####` with immutable hashes
5. **Lexical near-duplicate pass**

6. Build shingles (k=10), MinHash (m=200)
7. Compute approximate Jaccard
8. Form clusters at `J ≥ 0.85`; mark duplicates at `J ≥ 0.92` [29]
9. **Containment pass**
10. Compute containment scores
11. Flag “subset docs” at containment ≥ 0.90 [29]
12. **Topic modeling pass**
13. Train LDA with K grid; choose stable K
14. Compute topic similarity; cluster at 0.80 cosine threshold [31]
15. **Document graph build**
16. Build directed edges by explicit references, shared IDs, shared interface mentions
17. Compute PageRank centrality (α=0.85) [36]
18. **Candidate core statement extraction**
19. Run TextRank per document section to identify salient sentences
20. Apply MMR to select a non-redundant candidate set used for drafting OCF modules [43]
21. **OCF module synthesis**
22. Create module stubs (Module-ID, type, primitive mapping)
23. Convert extracted claims into typed statements with parameter definitions
24. Attach provenance links (source + location)
25. Identify dependencies and verification hooks [44]
26. **Lossless merge and traceability audit**
27. For each source statement, assign mapping status: {mapped exact, mapped partial, not mapped}
28. Achieve TCM ≥ 0.98 before any deprecations
29. **Derived view generation**
30. Generate role-specific views (AI training spec, internal dev guide, interface catalog)
31. Evaluate summaries with ROUGE against human gold summaries for consistency of derived views (secondary metric) [39]

32. **Deprecation decision and governance enforcement**

33. Produce deprecation matrix with scores and rationale
34. Move archive candidates to controlled archive; delete only after waiting period and dependency scan
35. Baseline OCF and enforce change control workflow [45]

Governance and anti-drift mechanisms

OCF remains stable only if the organization prevents re-fragmentation. Governance is anchored in requirements traceability and configuration/change control.

Requirements and traceability discipline. Requirements management includes establishing the organization and procedures (e.g., change board), maintaining bidirectional traceability, evaluating change requests to baselines, and maintaining consistency across requirements, ConOps, and architecture/design. [21] OCF implements this by requiring: no baseline edits without a change request; no new module without trace-back; no requirement without verification intent. [25]

Configuration management discipline. Configuration management controls changes made to baselines and tracks versions, ensuring the product configuration is known and consistent with the information describing the product. [22] OCF treats the module registry as configuration items: each module has unique ID; release packaging is governed; deltas are auditable. [45]

Change control with board workflow. The handbook describes a typical change control process and a configuration control board approach (receive/log → review → decision → execute → verify → release documentation → close request). [23] OCF adopts this structure for doc changes: changes are proposals; modules are baselined artifacts; decisions are recorded; releases are versioned. [26]

Workflow diagram for consolidation and governance (mermaid)

```
flowchart TD
  A[Inventory + Normalize Corpus] --> B[Near-duplicate + Containment Clustering]
  B --> C[Topic Modeling Clusters]
  C --> D[Doc Graph + Centrality (PageRank)]
  D --> E[Extract Candidates (TextRank)]
  E --> F[Select Non-Redundant Set (MMR)]
  F --> G[Draft OCF Modules + Provenance]
  G --> H[Trace Coverage Audit]
  H -->|TCM >= 0.98| I[Baseline OCF Release]
  H -->|TCM < 0.98| G
  I --> J[Deprecation Matrix + Archive/Delete Decisions]
  J --> K[Governed Change Control for Future Updates]
```

Actionable next-steps checklist

- Establish OCF repository structure (core modules, derived views, archive)
- Assign owners for: module registry, provenance mapping, verification hooks

- Run the pipeline through clustering + graph centrality
- Draft module stubs before writing prose (schema-first)
- Enforce traceability audit (TCM) gate before archiving any files
- Baseline OCF v1.0, then freeze legacy docs except via change control
- Start “no-new-doc-without-module-ID” rule on day one after baseline

Proposed schedule (Gantt-style mermaid timeline)

```
gantt
  title OCF Consolidation Program Schedule (Template)
  dateFormat  YYYY-MM-DD
  axisFormat  %b %d

  section Foundation
  Define scope, roles, and baselines            :a1, 2026-03-03, 5d
  Tooling setup (parsing, IDs, storage)         :a2, after a1, 5d

  section Corpus Analysis
  Inventory + normalization                     :b1, after a2, 10d
  Near-duplicate + containment clustering        :b2, after b1, 7d
  Topic modeling + thematic clustering           :b3, after b2, 7d
  Graph build + centrality scoring              :b4, after b3, 5d

  section Core Synthesis
  Candidate extraction (TextRank)               :c1, after b4, 5d
  MMR selection and module drafting             :c2, after c1, 10d
  Trace coverage audit + gap closure             :c3, after c2, 10d

  section Governance + Deprecation
  Baseline OCF v1.0 release                     :d1, after c3, 3d
  Deprecation matrix + archive migration         :d2, after d1, 7d
  Waiting period before deletions               :d3, after d2, 30d
  Final deletions + post-mortem                  :d4, after d3, 3d
```

Works Cited

Piotr Indyk [46] and Rajeev Motwani [47]. “Approximate Nearest Neighbors: Towards Removing the Curse of Dimensionality.” *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing (STOC ’98)*, ACM, 1998, pp. 604–613. doi:10.1145/276698.276876. [30]

Moses Charikar [48]. “Similarity Estimation Techniques from Rounding Algorithms.” *Proceedings of the Thirty-Fourth Annual ACM Symposium on Theory of Computing (STOC ’02)*, ACM, 2002, pp. 380–388. doi:10.1145/509907.509965. [49]

Andrei Z. Broder [50]. “On the Resemblance and Containment of Documents.” *Compression and Complexity of Sequences 1997 (SEQUENCES ’97)*, IEEE, 1997, pp. 21–29. doi:10.1109/SEQUEN.1997.666900. [29]

David M. Blei [51], Andrew Y. Ng [52], and Michael I. Jordan [53]. “Latent Dirichlet Allocation.” *Journal of Machine Learning Research*, vol. 3, 2003, pp. 993–1022. doi:10.1162/jmlr.2003.3.4-5.993. [31]

Jaime Carbonell [54] and Jade Goldstein [55]. “The Use of MMR, Diversity-Based Reranking for Reordering Documents and Producing Summaries.” *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR ’98)*, ACM, 1998. doi:10.1145/290941.291025. [56]

Rada Mihalcea [57] and Paul Tarau [58]. “TextRank: Bringing Order into Text.” *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing*, Association for Computational Linguistics, 2004, pp. 404–411. [34]

Chin-Yew Lin [59]. “ROUGE: A Package for Automatic Evaluation of Summaries.” *Text Summarization Branches Out*, Association for Computational Linguistics, 2004, pp. 74–81. [39]

Sergey Brin [60] and Lawrence Page [61]. “The Anatomy of a Large-Scale Hypertextual Web Search Engine.” *Computer Networks*, vol. 30, no. 1–7, 1998, pp. 107–117. doi:10.1016/S0169-7552(98)00110-X. [36]

Claude E. Shannon [62]. “A Mathematical Theory of Communication.” *Bell System Technical Journal*, vol. 27, 1948, pp. 379–423, 623–656. doi:10.1002/j.1538-7305.1948.tb01338.x. [63]

International Organization for Standardization [64]. *ISO/IEC/IEEE 42010:2011 Systems and Software Engineering—Architecture Description*. ISO, 2011. [13]

ISO/IEC/IEEE 15288:2015 Systems and Software Engineering—System Life Cycle Processes. ISO, 2015. [16]

ISO/IEC/IEEE 15289:2019 Systems and Software Engineering—Content of Life-Cycle Information Items (Documentation). ISO, 2019. [19]

Institute of Electrical and Electronics Engineers [65]. “ISO/IEC/IEEE 42010 (IEEE Listing).” IEEE Standards. [66]

“ISO/IEC/IEEE 15288 (IEEE Listing).” IEEE Standards. [67]

“ISO/IEC/IEEE 15289 (IEEE Listing).” IEEE Standards. [68]

National Aeronautics and Space Administration [9]. *NASA Systems Engineering Handbook (NASA/SP-2016-6105 Rev2).* 2016. [69]

Edward L. Deci [70] and Richard M. Ryan [71]. “The ‘What’ and ‘Why’ of Goal Pursuits: Human Needs and the Self-Determination of Behavior.” *Psychological Inquiry*, vol. 11, no. 4, 2000, pp. 227–268. doi:10.1207/S15327965PLI1104_01. [72]

Amy C. Edmondson [73]. “Psychological Safety and Learning Behavior in Work Teams.” *Administrative Science Quarterly*, vol. 44, no. 2, 1999, pp. 350–383. doi:10.2307/2666999. [74]

Pascal Fries [75]. “Rhythms for Cognition: Communication Through Coherence.” *Neuron*, vol. 88, no. 1, 2015, pp. 220–235. doi:10.1016/j.neuron.2015.09.034. [76]

Feedback Systems: An Introduction for Scientists and Engineers [77]. Caltech, updated PDF edition. [78]

Convex Optimization [79]. Cambridge University Press, 2004. [80]

Chemical Oscillations, Waves, and Turbulence [81]. Springer-Verlag, 1984. doi:10.1007/978-3-642-69689-3. [82]

Daniel Kahneman [83] and Amos Tversky [84]. “Prospect Theory: An Analysis of Decision under Risk.” *Econometrica*, vol. 47, no. 2, 1979, pp. 263–291. doi:10.2307/1914185. [85]

David Laibson [86]. “Golden Eggs and Hyperbolic Discounting.” *The Quarterly Journal of Economics*, vol. 112, no. 2, 1997, pp. 443–477. doi:10.1162/003355397555253. [87]

---

[1] [49] https://dl.acm.org/doi/10.1145/509907.509965
https://dl.acm.org/doi/10.1145/509907.509965

[2] [3] [11] [13] [14] [59] ISO/IEC/IEEE 42010:2011 - Architecture description
https://www.iso.org/standard/50508.html?utm_source=chatgpt.com

[4] [15] [16] [61] ISO/IEC/IEEE 15288:2015 - Systems and software ...
https://www.iso.org/standard/63711.html?utm_source=chatgpt.com

[5] [7] [8] [10] [17] [21] [22] [23] [25] [26] [38] [41] [45] [52] [54] [64] [69] [75] [86] https://www.nasa.gov/wp-content/uploads/2018/09/nasa_systems_engineering_handbook_0.pdf
https://www.nasa.gov/wp-content/uploads/2018/09/nasa_systems_engineering_handbook_0.pdf

[6] [9] [27] [29] [42] [46] [53] [70] [73] [79] [83] On the resemblance and containment of documents
https://www.misserpirat.dk/main/docs/00000004.pdf?utm_source=chatgpt.com

[12] [39] [58] [60] [65] https://aclanthology.org/W04-1013/
https://aclanthology.org/W04-1013/

[18] [19] [24] [40] [44] [48] [51] [71] ISO/IEC/IEEE 15289:2019 - Systems and software ...
https://www.iso.org/standard/74909.html?utm_source=chatgpt.com

[20] [57] [72] https://www.tandfonline.com/doi/abs/10.1207/S15327965PLI1104_01
https://www.tandfonline.com/doi/abs/10.1207/S15327965PLI1104_01

[28] Latent Dirichlet Allocation
https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf?utm_source=chatgpt.com

[30] https://dl.acm.org/doi/10.1145/276698.276876
https://dl.acm.org/doi/10.1145/276698.276876

[31] [50] https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf
https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf

[32] [77] https://pmc.ncbi.nlm.nih.gov/articles/PMC7298183/
https://pmc.ncbi.nlm.nih.gov/articles/PMC7298183/

[33] The Use of MMR, Diversity-Based Reranking for ...
https://www.cs.cmu.edu/~jgc/publication/The_Use_MMR_Diversity_Based_LTMIR_1998.pdf?utm_source=chatgpt.com

34 35 43 55 62 81 TextRank: Bringing Order into Text
https://aclanthology.org/W04-3252/?utm_source=chatgpt.com

36 37 https://www.sciencedirect.com/science/article/pii/S016975529800110X
https://www.sciencedirect.com/science/article/pii/S016975529800110X

47 68 IEEE/ISO/IEC 15289-2019
https://standards.ieee.org/ieee/15289/7196/?utm_source=chatgpt.com

56 The use of MMR, diversity-based reranking for reordering ...
https://dl.acm.org/doi/abs/10.1145/290941.291025?utm_source=chatgpt.com

63 https://onlinelibrary.wiley.com/doi/abs/10.1002/j.1538-7305.1948.tb01338.x
https://onlinelibrary.wiley.com/doi/abs/10.1002/j.1538-7305.1948.tb01338.x

66 IEEE/ISO/IEC 42010-2011
https://standards.ieee.org/ieee/42010/5334/?utm_source=chatgpt.com

67 IEEE/ISO/IEC 15288-2015
https://standards.ieee.org/ieee/15288/5673/?utm_source=chatgpt.com

74 https://journals.sagepub.com/doi/10.2307/2666999
https://journals.sagepub.com/doi/10.2307/2666999

76 https://pubmed.ncbi.nlm.nih.gov/26447583/
https://pubmed.ncbi.nlm.nih.gov/26447583/

78 https://www.cds.caltech.edu/~murray/books/AM08/pdf/am08-complete_22Feb09.pdf
https://www.cds.caltech.edu/~murray/books/AM08/pdf/am08-complete_22Feb09.pdf

80 https://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf
https://web.stanford.edu/~boyd/cvxbook/bv_cvxbook.pdf

82 https://discovery.researcher.life/search/article?
doi=10.1007%2F978-3-642-69689-3&secondary_source=oa.mg_button&utm_campaign=oa.mg_button&utm_medium=oa.mg_m
https://discovery.researcher.life/search/article?
doi=10.1007%2F978-3-642-69689-3&secondary_source=oa.mg_button&utm_campaign=oa.mg_button&utm_medium=oa.mg_medium&utm_source=oa.r

84 85 https://www.scirp.org/reference/referencespapers?referenceid=987257
https://www.scirp.org/reference/referencespapers?referenceid=987257

87 https://www.researchgate.net/publication/24091574_Golden_Eggs_and_Hyperbolic_Discounting
https://www.researchgate.net/publication/24091574_Golden_Eggs_and_Hyperbolic_Discounting