

TopicQuests

Conversence

Open Sherlock



Jack Park

TopicQuests Foundation

Marc-Antoine Parent

Conversence

Presentation: 20220624

© 2022 TopicQuests Foundation

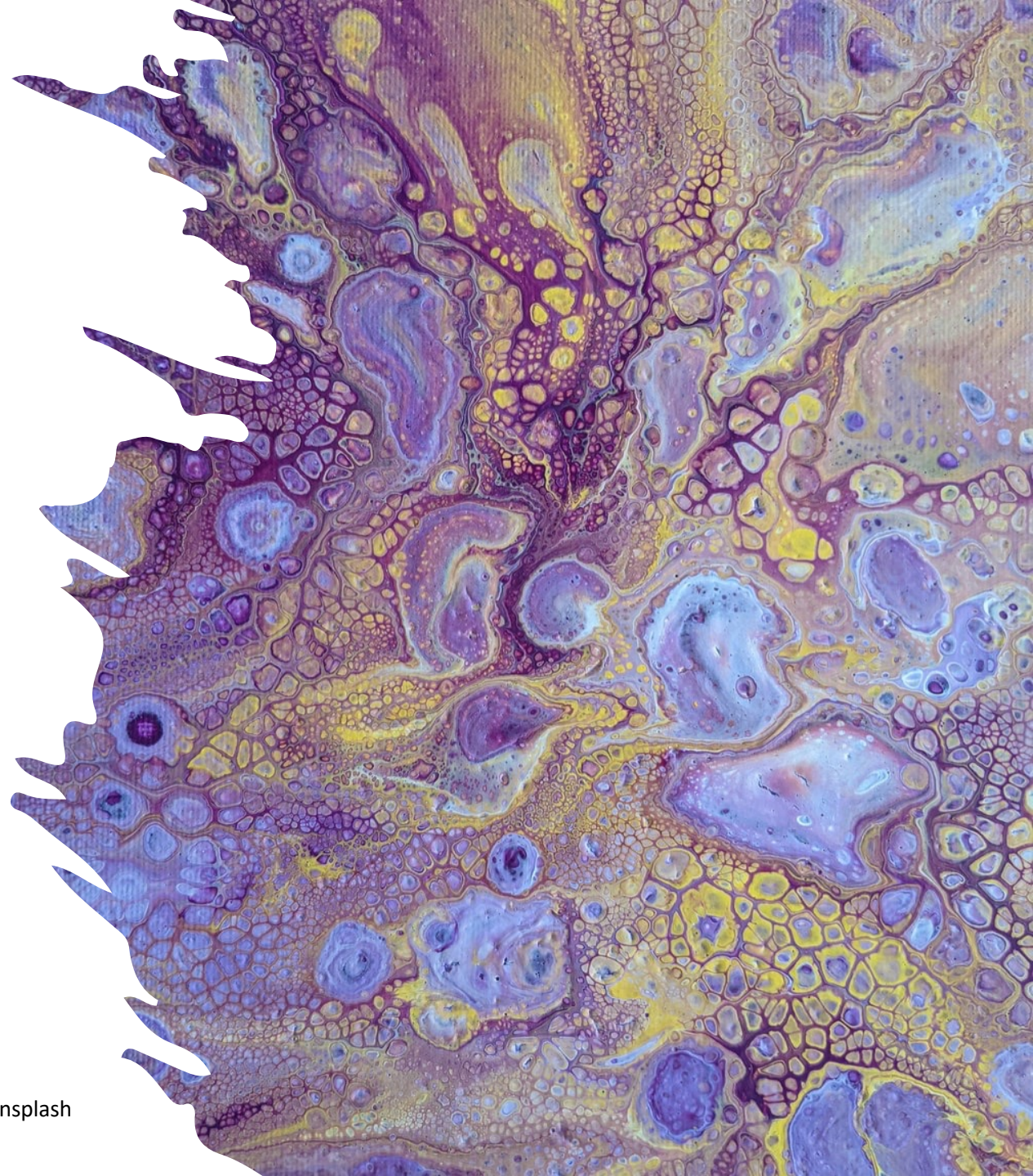


Jplenio <https://pixabay.com/photos/sunset-tree-dawn-sun-nature-dusk-3156440/>

# Context and problems on the table?

- Context
  - Improved human/machine systems capabilities in the face of complex, urgent problems (Engelbart)
- Problems
  - Human activity
    - Collaboration
    - Information Silos
  - Knowledge representation
  - User experience

Photo by Sigmund on Unsplash



# Research Questions

- Can a topic map *learn by reading*?
- What software architectures support machine reading in the context of topic mapping?

Photo by Elliott Reyna on Unsplash



# Solution research under way?

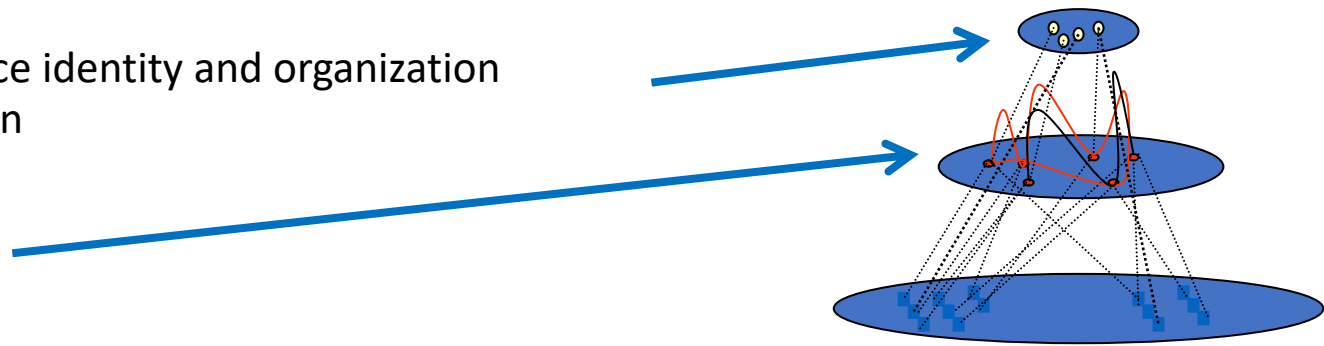
- Collaboration and User Experience
  - Knowledge gardens
    - Subject of a different talk
- Open Source: OpenSherlock
  - Information Silos
    - Literature-based discovery
    - Machine reading
    - Knowledge federation
  - Knowledge representation
    - A range of representation systems working together
      - Hybrid architectures
      - Pandemonium-style architectures



Photo by Kaleidico on Unsplash

# Towards a “Cognitive” Platform: OpenSherlock\*

- OpenSherlock is:
  - A Topic Map for information resource identity and organization
  - A *society of agents* system which can
    - Read documents
    - Process information resources
      - Maintain the WordGram graph
      - Maintain the topic map
      - Build and maintain models
      - Perform discovery tasks
        - Read Literature
        - Physical process discovery
        - Literature-based discovery
        - (eventually) Theory formation
      - (eventually) Answer questions
  - Agents are coordinated by:
    - A blackboard system
    - A dynamic task-based agenda
    - Event propagation and handling
      - Stream processing



\*OpenSherlock is an open source mostly Java-based research platform.

# OpenSherlock's Use Cases

- 1) Merge decision and implementation for topic maps
- 2) Research: machine reading
- 3) Literature-based discovery across the entire topic map and beyond
- 4) (experimental) Question answering
- 5) (eventually) Theory formation, process discovery

# A Cognitive Architecture

- “A cognitive architecture is, first, a software architecture that supports communication between software modules, shared resources and workspaces, and control. What makes it cognitive is that the modules implement abilities we generally call perceptual or cognitive.”†
  - Long-term Memory (LTM)
  - Short-term (working) Memory (STM)
  - Pattern recognition
  - Symbol manipulation
  - Decision making
  - Data
  - ...



Photo by Michal Vrba on Unsplash

†Cohen, P (2010) *Lecture 3: Cognitive Architecture*.

<http://w3.sista.arizona.edu/~cohen/cs-665-spring-2010/lectures/PDFs/Lecture3.pdf>

# Machine Reading: Natural Language “Understanding”

- General approach
  - Engage
    - Lexicon / ontology
    - Parser
    - Grammar rules
- Our approach
  - Engage
    - Lexicon / ontology (*topic map*)
    - Templates
    - Lenses
    - Grammar rules
    - Machine Learning and Parsing

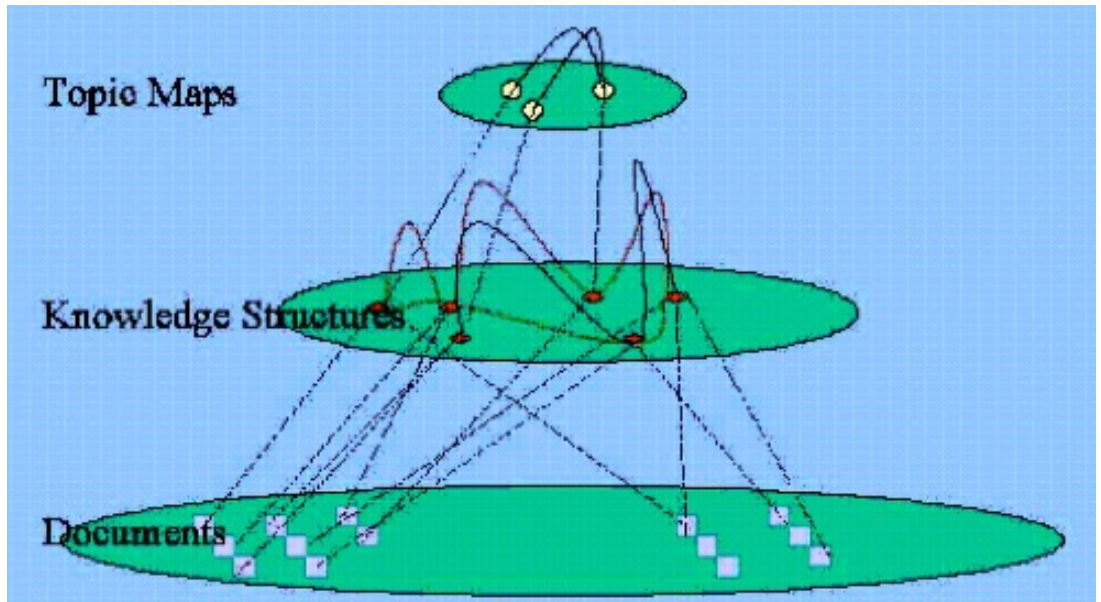
Photo by Bethany Laird on Unsplash





# Towards a Machine Reading Architecture

- Workflow
  - Documents → Intermediate Structures → Higher-order Structures
    - TopicMaps
    - Conceptual Graphs
    - Probabilistic Graphs
    - ...



# Topic Maps: Long Term Memory (LTM)

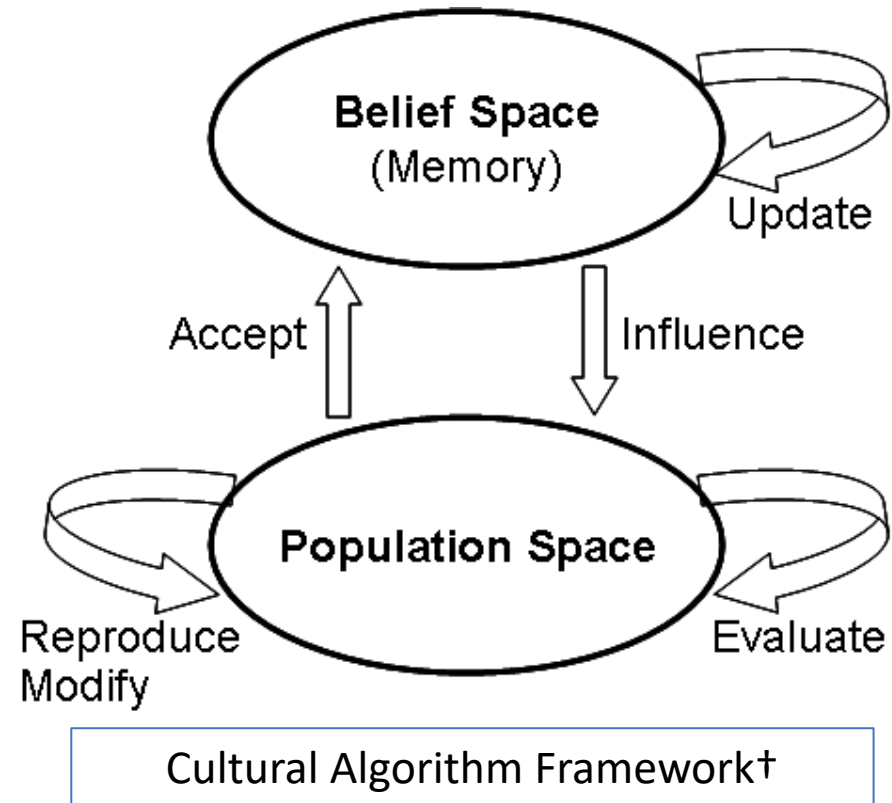
- A Topic Map is like a library without all the books
  - A container for a universe of topics
  - A Topic Map is *indexical*
    - Like a card catalog
      - Each topic has its own representation (proxy)
    - Improving on a card catalog, a topic can be identified many different ways
    - Captures metadata and optionally content
  - A Topic Map is *relational*
    - Like a good road map
      - Topics are connected by associations (relations)
      - Topics point to their occurrences in the *territory*
  - A Topic Map is *organized*
    - Multiple records on the same topic are *co-located* (stored as one topic) in the map
      - One location in the map for each topic



Img: Jack Park

# Observations 1

- A Topic Map is central to the key research question
  - It serves as a kind of *memory* for social processes
  - It provides a robust platform for *subject identity*
  - It also serves as a repository for domain-specific vocabularies (ontologies, taxonomies, naming conventions,...)
    - It federates those vocabularies



† After Figure 1: Reynolds, RG & Peng, B (2005) CULTURAL ALGORITHMS: COMPUTATIONAL MODELING OF HOW CULTURES LEARN TO SOLVE PROBLEMS: AN ENGINEERING EXAMPLE. *Cybernetics and Systems An International Journal* Volume 36, 2005 - Issue 8

# Observations 2

- A Topic Map is *necessary* but *not sufficient* to support discovery, learning, or problem solving
  - It provides a powerful *indexical* structure related to the key artifacts in any universe of discourse:
    - Actors
    - Their relations
    - Their states
    - Rules, laws, theories,...
- To model those key artifacts, other representation strategies are required, which layer above the Topic Map
  - Conceptual Graphs
  - Qualitative Process Theory
  - Belief Networks
  - Deep Learning
  - ...

# Knowledge Representation: Symbol Systems

- A physical symbol system (also called a formal system) takes physical patterns (symbols), combining them into structures (expressions) and manipulating them (using processes) to produce new expressions.†
  - We consider the *identifier* of a topic in a topic map *as* a symbol
  - Example base structure: RDF triples
    - { subject symbol, predicate symbol, object symbol }

† [https://en.wikipedia.org/wiki/Physical\\_symbol\\_system](https://en.wikipedia.org/wiki/Physical_symbol_system)

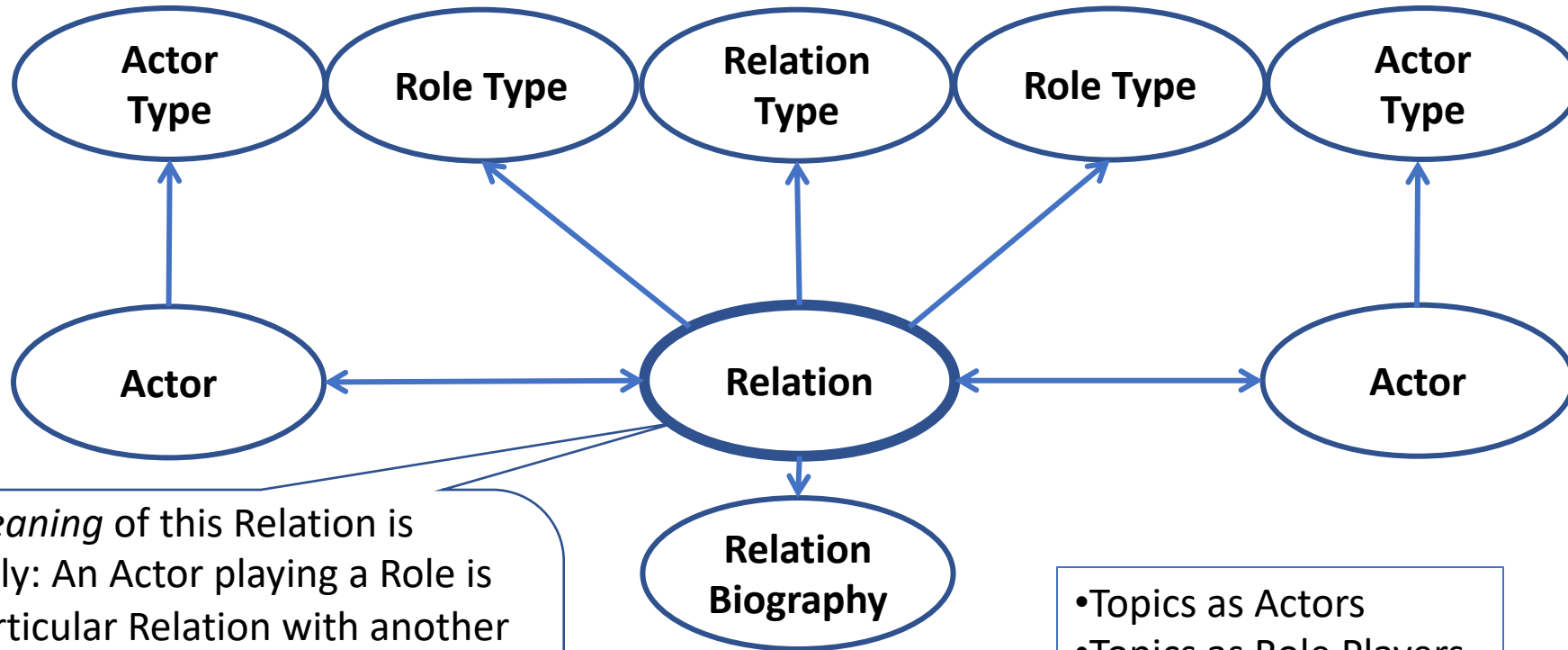
# asA?

- For taxonomy, we have *isA*
  - And its variants
    - subclassOf
    - instanceOf
- For roles, we do not have *asA*
  - Instead, we model *roles*
  - Ulam quote

"When you perceive intelligently, you always perceive a function, never an object in the physical sense. Cameras always register objects, but human perception is always the perception of functional roles. The two processes could not be more different.... Your friends in AI are now beginning to trumpet the role of contexts, but they are not practicing their lesson. They still want to build machines that see by imitating cameras, perhaps with some feedback thrown in. Such an approach is bound to fail..."

Stanislaw Ulam † :

# TopicMap Structure



The *meaning* of this Relation is precisely: An Actor playing a Role is in a particular Relation with another Actor playing a Role, in the context of biographical records.

Note: Since this Relation is a Topic, it, too, can be an Actor in other Relations

- Topics as Actors
- Topics as Role Players
- Topics as Relations
- Topics as Types
- Topics as Biographies

# Context, Scopes

- Let us consider a *claim* in the form of a topic relation
- When a claim is asserted, that claim is surrounded by context
- We can say that every topic has a *biography*
  - A biography can include
    - Historical records
    - Specific dates
    - Experiments
    - Justifications and objections
    - Epistemic status
    - Occurrences



# Shifting to Implementation Concepts

# Machine Reading in OpenSherlock

- Process Loop:

- For a given document

Each Document has a  
Blackboard (STM)

- For every paragraph in that document

Each Paragraph has a  
Blackboard (STM)

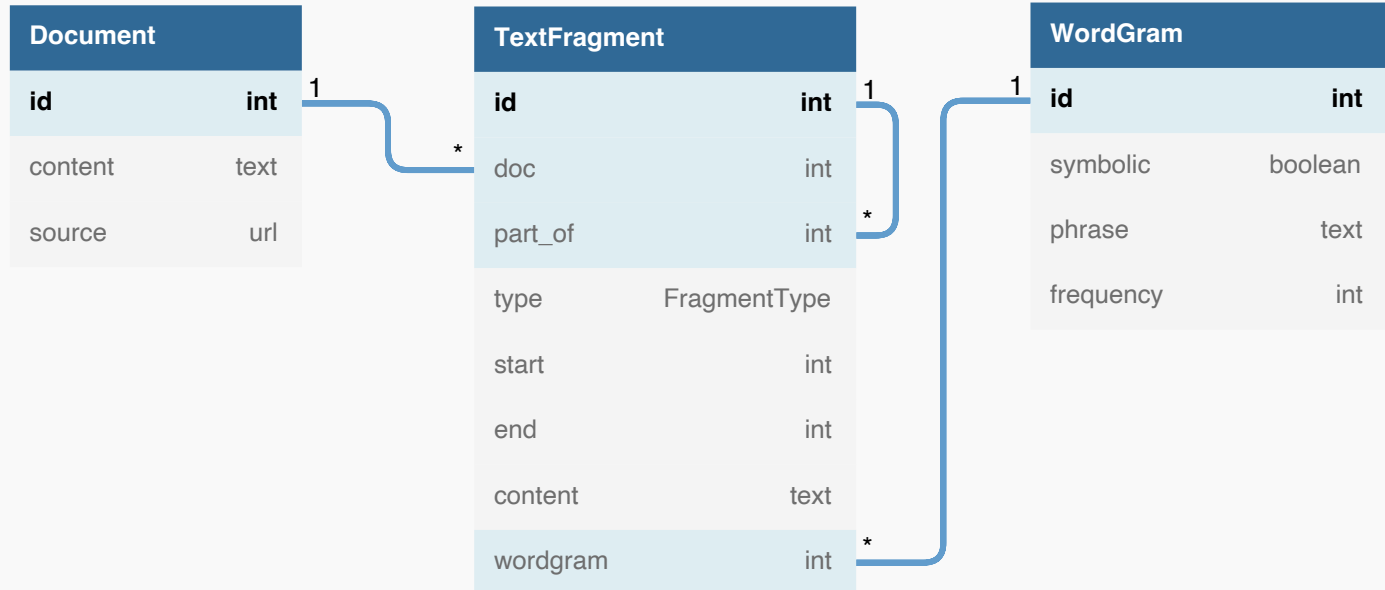
- For every sentence in each paragraph

- *Read* the sentence

Each Sentence has a  
Blackboard (STM)

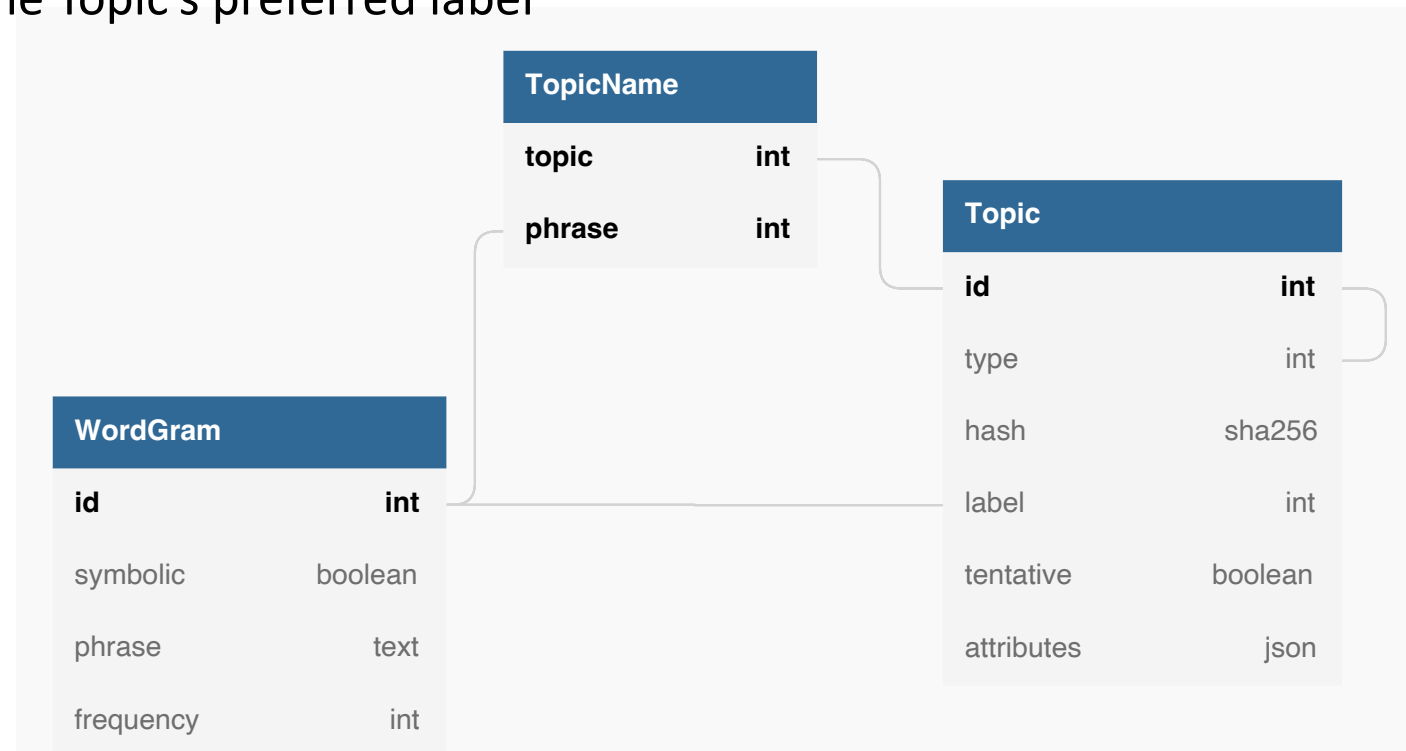
# Sentence Reading

- First Step: Identify common phrases (and words) in sentences
  - The phrase's occurrence at a location in a document is a TextFragment
  - WordGram structure records aggregate metadata about recurring phrases
    - Count of TextFragments serves as basis for probabilistic models



# Sentence Reading

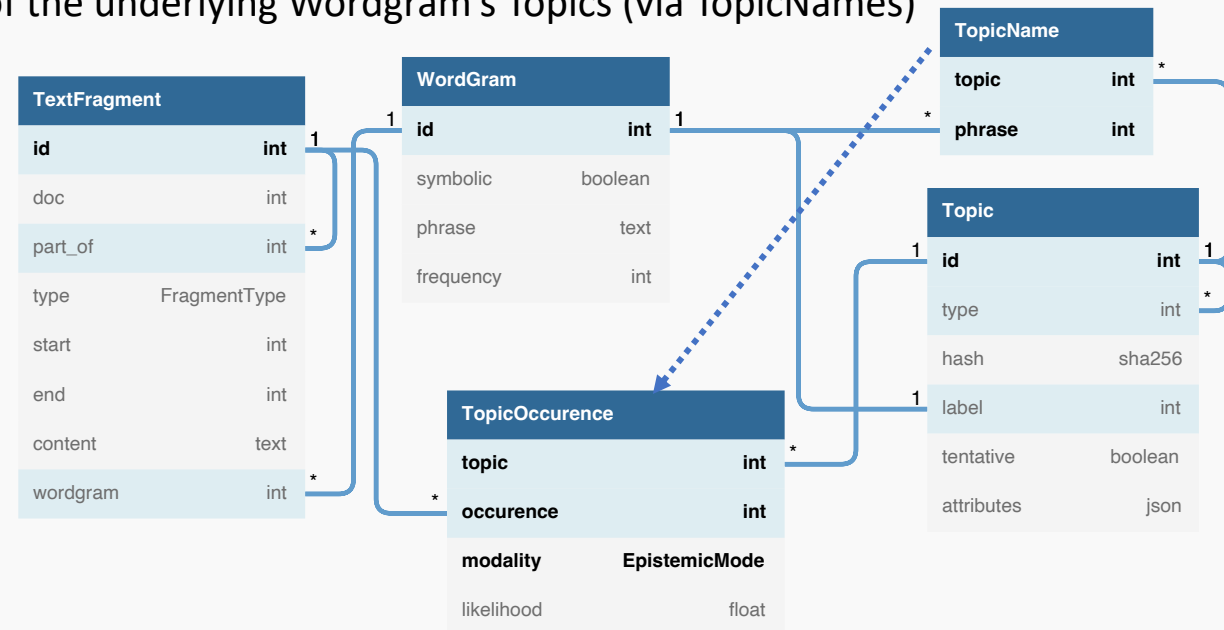
- Second Step:
  - Frequent WordGrams may already have been associated to Topics (TopicName relation)
    - A single name is designated as the Topic's preferred label



# Sentence Reading

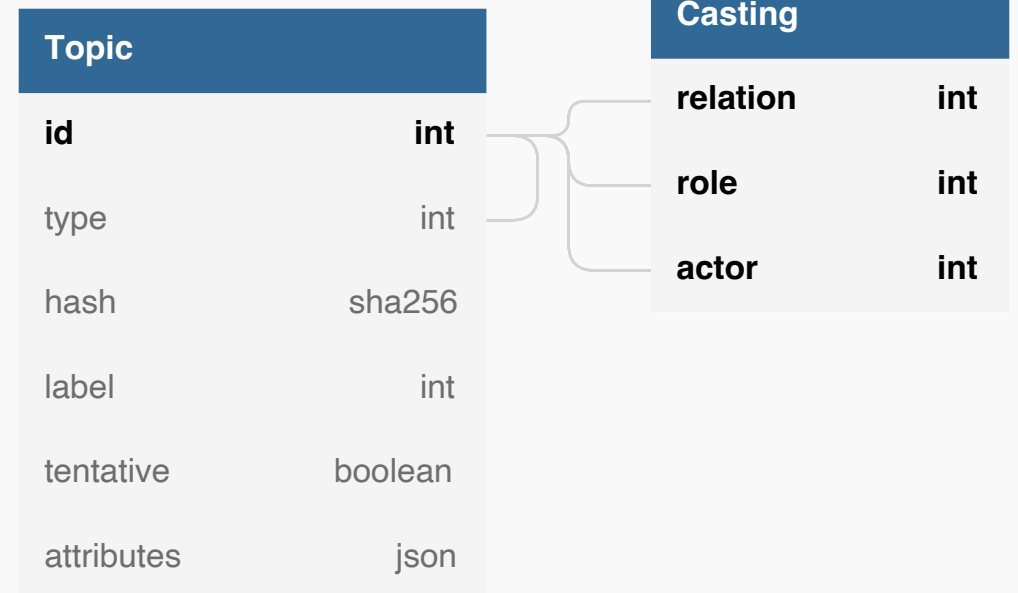
- Second Step: According to how many Topics are found through TopicNames:
  - Single Topic: Create a TopicOccurrence, associating the TextFragment to the Topic
  - If no Topic yet exists for a frequent WordGram, create a tentative Topic placeholder
  - Multiple Topics: See if the context allows to disambiguate.
  - Multiple plausible Topics: Create many tentative TopicOccurrences, in order of plausibility.

The TextFragment's Topics (via TopicOccurrences) are a subset of the underlying Wordgram's Topics (via TopicNames)

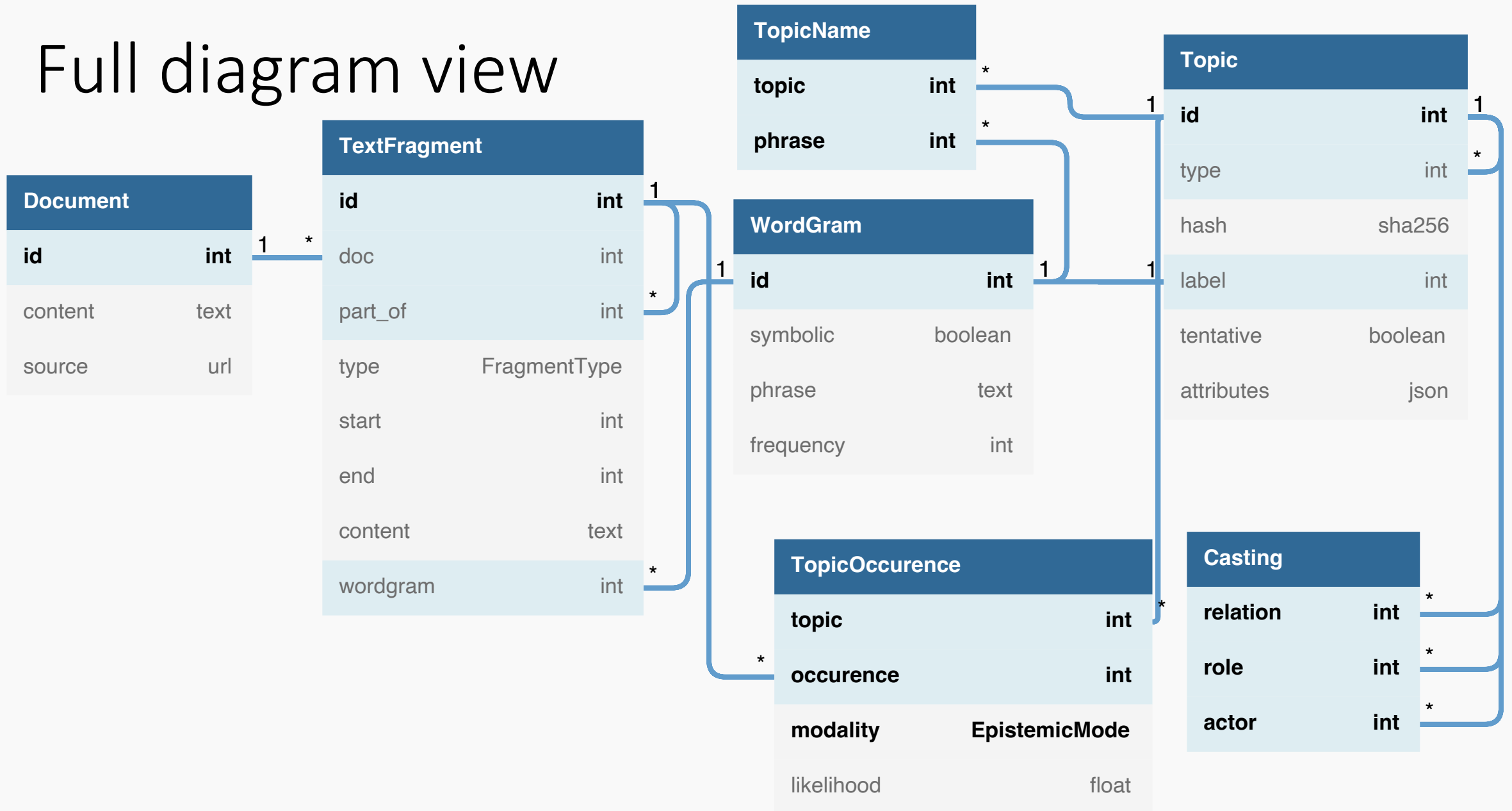


# Sentence Reading

- Second Step:
  - If the Topic's type is a known RelationType, identify the other component Topics (Actors) and their Roles. Store those Castings in the relational Topic.
    - This is similar to a Frame (FrameNet)
  - Continue recursively, as Relations can be Actors in higher-order Relations (e.g. causal inference, evidence, etc.)



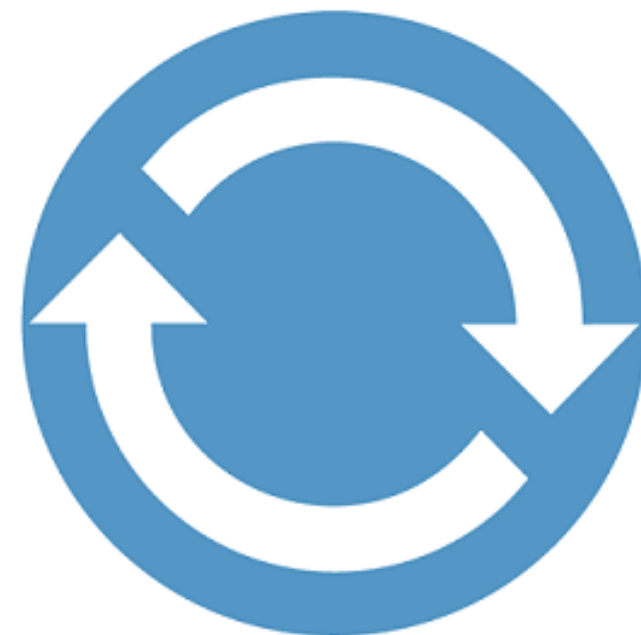
# Full diagram view



# Sentence Reading as an Iterative Process

---

- Since the same WordGrams serve many sentences
  - Any change of interpretation of a WordGram (TopicName relations) affects TopicOccurrences
    - Internal: Change from *Unknown* to new *Noun*, *Noun Phrase*, *Verb* or *Verb Phrase*
    - Creation of a tentative Topic
    - Tentative Topic becomes well-defined
    - Association with a previously defined Topic
  - TopicName change events enable unfinished (not fully read) sentences to iterate and possibly finish.





# Sentence Reading Revisited

- Sometimes
  - TopicMap *recognizes* terms
- Mostly
  - Sentence is read by pattern recognition tools
    - IF-THEN sentence structure rules (**Grammar rules**)
      - Augmented by *Lenses*
    - Frame Semantics (e.g., FrameNet)<sup>†</sup> (**Templates**)
    - Language Models
  - TopicMap *learns* from those results

<sup>†</sup> Baker, CF, Fillmore, CJ, Lowe, JB (1998). The Berkeley Framenet Project. *COLING '98 Proceedings of the 17th international conference on Computational linguistics*, Volume 1

# Lenses

- Lenses are referenced in specific WordGrams. When a WordGram is recognized in a sentence, its lens, if any, is activated
- Domain-specific interpreters which detect structures
  - Taxonomy
  - Causality
  - Biomedical
  - Geophysical
  - ...
- Can include
  - Rules
  - Conceptual Graphs
  - Deep learning
  - Process models
    - Built and maintained while reading
    - Predict while reading – *Anticipatory Reading*
  - ...



# From WordGrams to relational Topics

- Identify elementary topics
  - Nouns: “CO2”, “Carbon Dioxide”, ... → [CO2]
  - Verbs: “causes”, “is caused by”, ... → [causes (cause, effect)]
- Construct Relation
  - Two sentence example
    - “CO2 is a cause of climate change.”
    - “Changing climate is caused by carbon dioxide.”
  - Both result in the same structure:
    - [causes (cause: [CO2], effect: [climate change])]

# Topic structure

- A Topic is a structured record of
  - Unique identifier
  - Topic type (or types?)
  - Attributes (key-value pairs, where the value is a literal)
  - Castings (key-value pairs, where the key is a role and the value is a topic)
  - The keys are elementary topics
  - The topic should be identified by its content, and should be content-addressable
    - Identifiers could be hashes?
- Each topic has a biography, the set of its TopicOccurrences

# Topic mapping process

- Disambiguation of subjects is a *topic mapping* process
  - *Learning* means continuous refinement of subject identity
    - Topics may be merged or split as we learn about their defining characteristics
    - We may define some of the Topic's attributes as unique for a given topic type and application domain
  - Ambiguities can also be solved through machine learning and/or human intervention

# A Simple Example

- Read this sentence:
  - **Gene expression is caused by soluble hormones binding to a plasma membrane hormone receptor**
- Topic Map recognizes:
  - “Gene expression” ← [GeneExpression]
  - “soluble hormones” ← [SolubleHormone]
  - “plasma membrane hormone receptor” ← [PlasmaMembraneReceptor]
- Software agents transform:
  - “is caused by” ← [Cause]
  - “binding to” ← [Binds]
- Final *semantic structure* (Nested N-Tuples):
  - [Cause  
  cause: [Binds what: [SolubleHormone] to: [PlasmaMembraneReceptor]]  
  effect: [GeneExpression]]

# A Complex Example

- The Sentence:
  - “The pandemic of obesity, type 2 diabetes mellitus (T2DM) and nonalcoholic fatty liver disease (NAFLD) has frequently been associated with dietary intake of saturated fats (1) and specifically with dietary palm oil (PO) (2).”
- Expected WordGram Triples:
  - [associated what: [Obesity] what: [saturated fats]]
  - [associated what: [Obesity] what: [palm oil]]
  - [associated what: [T2DM] what: [saturated fats]]
  - [associated what: [T2DM] what: [palm oil]]
  - [associated what: [NAFLD] what: [saturated fats]]
  - [associated what: [NAFLD] what: [palm oil]]

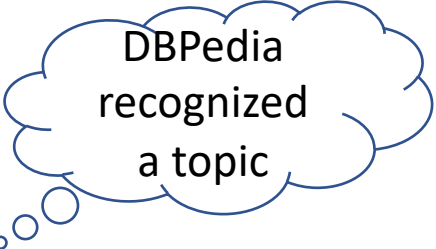
# Just One Triple—In 3 Slides—The Predicate

```
{  
  "pred": {  
    "gramSize": "2",  
    "sentences": ["ef471f0c-08c9-4097-ac17-354a32944fe1"],  
    "vers": "1489448293804",  
    "words": "associated with",  
    "Lex Types": ["vp"],  
    "predicate Tense": "present",  
    "id": "27637.27587",  
    "gramType": "pair",  
    "lensCodes": ["BioLens"]  
  },  
}
```

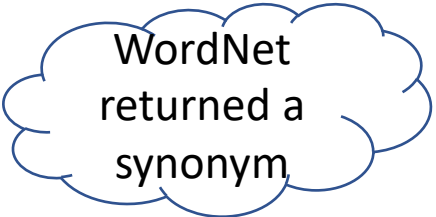


# The Object

```
"obj": {  
  "dbpo": {  
    "@percentageOfSecondRank": "0.0",  
    "@URI": "http://dbpedia.org/resource/Saturated_fat",  
    "@support": "455",  
    "@surfaceForm": "saturated fats",  
    "@offset": "156",  
    "@similarityScore": "1.0",  
    "@types": ""  
  },  
  "gramSize": "2",  
  "sentences": ["ef471f0c-08c9-4097-ac17-354a32944fe1"],  
  "synonyms": ["27739."],  
  "vers": "1489448293876",  
  "words": "saturated fats",  
  "lexTypes": ["np"],  
  "id": "27738.13882",  
  "gramType": "pair"  
},
```



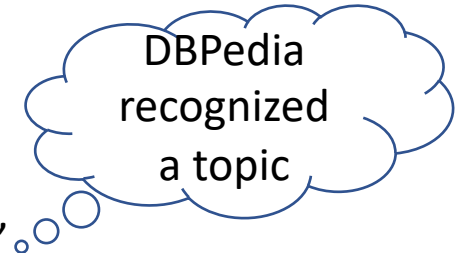
DBpedia  
recognized  
a topic



WordNet  
returned a  
synonym

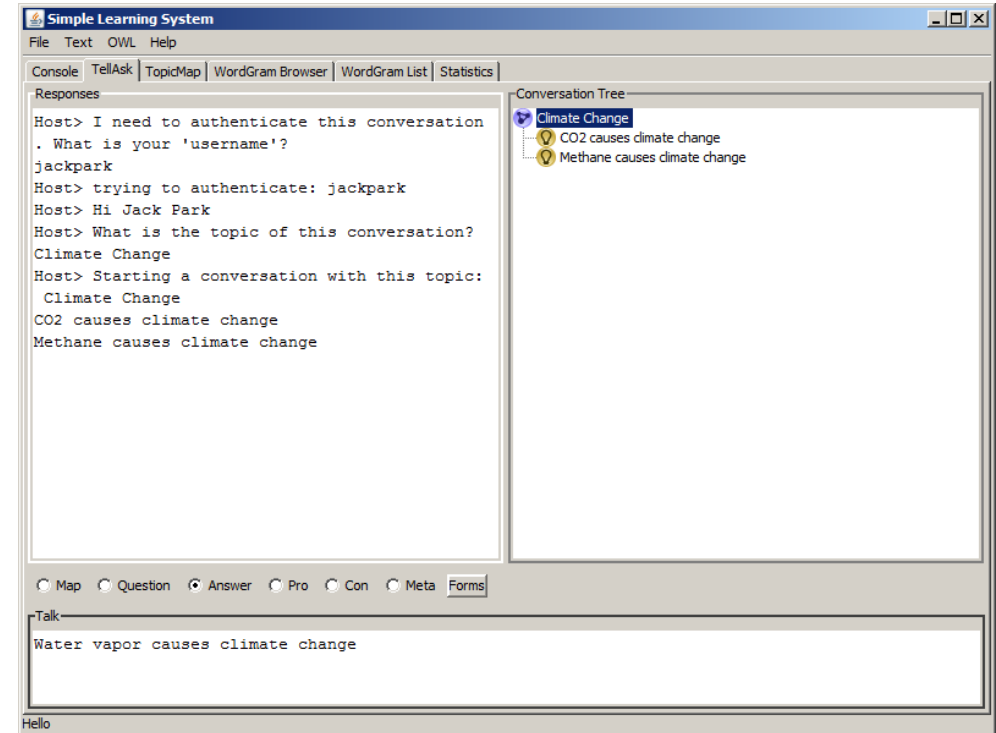
# The Subject and WordGram Triple Identity

```
"id": "27000._27637.27587_27738.13882T",
"subj": {
  "dbpo": {
    "@percentageOfSecondRank": "1.0799338377810428E-21",
    "@URI": "http://dbpedia.org/resource/Obesity",
    "@support": "3012",
    "@surfaceForm": "obesity",
    "@offset": "16",
    "@similarityScore": "1.0",
    "@types": "DBpedia:Disease"
  },
  "gramSize": "1",
  "sentences": ["ef471f0c-08c9-4097-ac17-354a32944fe1"],
  "vers": "1489448293875",
  "words": "obesity",
  "lexTypes": ["n"],
  "id": "27000.",
  "gramType": "singleton"
}
```



# Current State of OpenSherlock

- In two words: Starting over.
- In the past -1:
  - Implemented on Apache Solr
  - Moved to ElasticSearch
- In the past -2:
  - Exclusively used grammar rules for parsing
    - Sentences in this presentation parsed with those
- Now using the spaCy ecosystem, and Ontologies as well as grammar rules
- Now moving to PostgreSQL



TellAsk Interface on the Solr version of OpenSherlock (SolrSherlock)