* StratML. SemanticWebAlignmentTheory. Tryton.
* DCI, Qi4j, FP, Node, Angular, Vert.x, ServiceMix. RDF4J. Neo4j (ML). HAL, OpenAPI, GraphQL. DIDs.
* Upper Ontology: [Schema.org](http://schema.org), ontologies, NLP (DBPedia). Proof of concept (d2rq, OpenAPI, GraphQL). Template, Kinds Grammar. URN Hashing. Quads Pairs Segments. Augmentations (Sails): CIDs (services), GroupID: Aggregator expression.
* CoSQL. Monadic Parser Combinators. Content addressable RDF (S-Expressions) DIDs URNs. Zippers.
* Hashing: Common Metamodel. From source URIs. Display Name (URNs in context / resolution). Layers: Graph layout. Progressive / hierarchical content hashing (contexts / assertions / traversals aggregation).
* Ordering / Aggregation functions / relations inference. Hashing: octal prev/curr/next relations traversal graph encoding.
* Hashing: Sets, Groups, Relations: a(b, c), Operations: p(a, b): c, Functions: f(a, ...): domain / range. Monads. Encoding. Graph Layout. Lattices (faceted / scaling property traversal).
* Map Reduce / Graph ML: Hashing. IDs: contextual / evolveable (monad). URNs: REST message driven (HATEOAS) endpoints. Functional aggregated (monads map) nodes traversal. Resource wrapper Monad API (browse: functional domain / range, layer types / layers kinds, signatures).
* Hashing: DIDs, CANs, CAMs, DLTs (Events). Content negotiation: signatures, headers (referrer: state browsing, E-Tag: state hash, facets: URN Dimensions).
* Relationship: PK of SKs, OKs, roles (metaclass of S/K occurrences). N-ary relations. Hashing: masks: traversal / functions / relations / operations (ternary CAM, 2 bits XOR: lattice encodings).
* Serialization:
* DagCBOR. XML (Beans, Externalizable): Functional Roles (metaclass, etc) Functional Transforms (Aggregation Templates). HAL / JSON: Functional Fields (codat / data flow).
* TMRM. Core / Representation Properties. Data Model.
* Messages: Command, Event : Fact (async / topic / bus / reactive / aggregator), Query. Streams / Subscriptions: Functional / Data flow signatures.
* Event Log: Persistence / Augmentations. Entity event (key / value) map / reduce (aggregation). URNs / DIDs payload.
* Stream processing: is a very flexible way of processing data using data pipelines. Many users are aggregating, enriching, and transforming data into new topics. It is a very quick and convenient way to process all data in real-time.
* Event sourcing: is a system design in which immutable events are stored as a single source of truth about the system. A typical use case for event sourcing can be found in bank systems when we are loading the history of transactions. The transaction is represented by an immutable event that contains all data describing what exactly happened in our account.
* Commit Log: Distributed Transactions.
* Functional Programming, Set Theory and Semantic Web
* Edit Function returns to a Browser.
* ISO 15926 / 13250 And Semantic Web AI / KR for dummies
* A Less Ephemeral Web. Annotated breadcrumbs. Index, referrer, session purpose tagged items search. Augmented browsing. Apache Stanbol.
* Fwd: RDF graph serialization as bytes: A solved problem?
* Browser: Activation. URNs type handler services (email, content types, [schema.org](http://schema.org)).