* Semiotics: DCI / MVC Augmentations / Dataflow:
* Upper Layers (hierarchical / contextual) domain driven entities / gestures / forms APIs.
* Data:
* MVC: Model.
* DCI: Data.
* Graph: Nodes.
* Grammar: Terminal. Object.
* Functional: Monads wrapped Values.
* Augmentation: Aggregation (Data Matching).
* Component: Index. CDI CI Resolved bindings: Schema / Behavior Services Interfaces.
* Protocol: CRUD. Semantic Relations Queries. (e.g.: same as, data matching). Augmentations.
* Entities: Nodes. Data. Resources. Models / State. Arcs Recognition (Behavior). Objects From Schema. (Concept) to Behavior (Sign) Mappings. Model Resource Monads Values / Transforms.
* Schema:
* MVC: View.
* DCI: Context.
* Graph: Type / Context (Attributes).
* Grammar: Non Terminal. Concept.
* Functional: Monads wrapped Value Types.
* Augmentation: Activation (Schema Matching).
* Component: Registry. CDI DI Resolved bindings: Data / Behavior Services Interfaces.
* Protocol: Forms. Dataflow. Functional Flows: Initial / Contextual Form, Transforms (Graph Naming domain / range) arcs population (Index) Naming gives next populated state Form (Index retrieved state).
* Entities: Type / Context. Schema. Kinds / Roles. Node Recognition (Data). Concepts From Behavior (Sign) to Data (Object) Mappings. Domain Resource Monads Values / Transforms.
* Behavior:
* MVC: Controller.
* DCI: Interaction.
* Graph: Arcs.
* Grammar: Rules / Productions. Sign (recursion).
* Functional: Monads Transforms.
* Augmentation: Alignment (Behavior Matching).
* Component: Naming. CDI DC Resolved bindings: Data / Schema Services Interfaces.
* Protocol: IDs. Semantic Models (Graph, Dimensional, Discrete) Reactive Endpoints Encoding. Graph Name Flows Resolution / Entailments. Hierarchical Graphs Order Relations. Functional / Grammars Contextual (Transforms) state flows.
* Entities: Arcs. Behavior. Statements Kinds / NER Aggregation. Context Types / Relationships / Dimensions Recognition (Schema). Signs From Data (Object) to Schema (Concept) Mappings. Model / Domain Resource Monads Transforms Dataflow: Fixtures.
* Functional Mappings Aggregation / Transforms Flows:
* (Node : Object, Type : Concept, Arc : Sign, Node : Object);
* (Node : Object, Type : Concept) : Arc : Sign;
* (Arc : Sign, Node : Object) : Type : Concept;
* (Type : Concept, Arc : Sign) : Object : Node;
* Functional Protocol: Dataflow / Parsing. Monads Wrapper / Wrapped Types / Values: Sign, Concept, Object Monads Contexts / Transforms Mappings. MVC / DCI Graph (Node, Type / Context, Arc, Node); Discrete / Continuous Models CSPOs Monads / Transforms. Model driven (Types Domain / Range ordered contexts) Monads / Transforms (Augmentations).
* Semiotics: Augmentations / Alignments Dataflow.
* Data: Semantics. Matching / IDs. Aggregation. Contexts / Occurrences. Dimensional Model. Dimensional Data Measures Relations (e.g.: same as, data matching).
* Index. Sessions. Protocol Bindings. DCI Data (CI Resolved). Nodes.
* Schema: Syntax / Grammar. Parsing / NLP / NER. Activation. Available Roles / Types / Kinds in Model Contexts. Discrete Model. Relationship Relation Roles (Entities Kinds, schema matching).
* Naming. Resolvable Available Interaction Mappings. Production Rules Matching. DCI Context.
* Behavior: Pragmatics. Protocols. Alignment. State: Nodes / Arcs Attributes / Values. Graph Model Contexts Nodes (Hierarchical Graph Order Relations, behavior matching: order cause / effects, CEP, entailments, recognitions).
* Registry. Role Bindings. Resolvable Productions (Dataflow). DCI Interactions (DC Aligned). Arcs.
* DCI Models Encodings:
* (Occurrence, Context, Aggregated Type Attributes, Attributes Values : Occurrences);
* Graph Model: (Node : Data, Type : Context, Arc : Interaction, Node : Type Value);
* Dimensional Model: (Measure, Dimension, Unit, Value : Measure);
* Discrete Model: (Relation, Relationship, Kind, Entity : Relation);
* Labeled Property Graph: Arc : Node.
* Graph Model Dimensional / Discrete Models axes hierarchicaly ordered Measures / Relations relations Dataflow.
* Functional APIs:
* Model Monads: Models CSPOs Wrapper / Wrapped Types / Values. Layers Contexts. Augmentations.
* Model Transforms: Layers Query / Selectors / CUD: Append versioned Statements to Models given available browsing state interactions rules.
* Domain Monads: DOM (Dynamic Object Model) Context Instances (Model Values) Entities.
* Domain Transforms: DOM Values Query / Selectors / CUD: Append versioned Statements to Models given available browsing state interactions rules.
* Co relations (Algebras / CoSQL / NoSQL / LInQ):
* Models. Layers: Augmentation steps reification. Layers Statements Roles.
* Base Model:
* (Context, Occurrence / Statement, Attribute / Kind, Value / Entity);
* Discrete Layers Model: (Relationships / Assertions). Continuos CEP: 0 Duration Measures / Axis Events Relations.
* (Relationship, Relation, Kind, Entity);
* Context, Roles, Causal, etc. Relations.
* Continuous Layers Model: (Dimensions / Measures).  Discrete CEP: Order / Containment Relations.
* (Dimension, Measure, Unit, Value);
* Distance, Equivalence, Transportation, etc. Relations.
* Functional DCI: Monads / Functors / Data.
* Wrapper Types (Data Roles). Encoding.
* Wrapped types (Data Values). Encoding.
* Functors: Dataflow Domain / Range specifications (Contexts). Encoding.
* Monads: Dataflow implementations (Interactions). Encoding.
* Functional Roles. Wrapped Models Resources Monads Wrappers Encoding (Dataflow Entities):
* DCI Data (ETL / Ontology Mappings). Layers: Augmentation steps reification. Layers Statements Roles.
* Class / Metaclass
* Role / Occurrence
* Value / Instance
* Attribute / Context
* Arcs: (Instance, Occurrence / Node Class, Attribute, Value); Data.
* Nodes: (Class, Metaclass, Context, Role); Contexts.
* Mappings: Models. Interactions.
* Layers Hierarchy: Role : Class; Value : Instance. Property Graphs.
* Layers:
* Arcs: (Role, Instance, Metaclass, Context); DCI Interactions.
* (Context, Role, Instance, Class);
* (Class, Context, Role, Instance);
* Nodes: (Instance, Class, Context, Role); DCI Contexts.
* Type: Class / Metaclass IDs
* Kind: Role / Occurrence IDs
* Node: Value / Instance IDs
* Arc: Attribute / Context IDs
* Layers:
* Arcs: (Arc, Node, Type, Kind); DCI Interactions.
* (Kind, Arc, Node, Type);
* (Type, Kind, Arc, Node);
* Nodes: (Node, Type, Kind, Arc); DCI Contexts.
* Models / Encoding:
* Layer Roles: (Context, Subject, Predicate, Object);
* Context: Kind Statements.
* Subject: Node Statements.
* Predicate: Arc Statements.
* Object: Node Statements.
* Type: Matching / Inferences.
* Encodings:
* Models: Quads / Property Graphs. DCI / Dataflow.
* Lists Model. Roles / Order.
* Hierarchical Graph Encoding.
* Sets.
* Functional DCI / Layers abstraction.
* Functional Parser. CUD. (data) of Grammar (Functional scheme) Dataflow Entities.
* Functional Parser. Grammar (Functional scheme) Dataflow Entities.
* Functional Parser. Parse (executions / behavior). Dataflow interactions instances.
* Codat: Dataflow / Protocol Prompts (run at).
* Protocol:
* Stateful I/O (ordered contexts). Reactive encoded Message driven gestures (CQRP).
* CDI / ESB Runtime. Backends. Connectors. CAM / CAN / DIDs.
* Dataflow HATEOAS.
* Augmentations / Alignments:
* Aggregation. Contexts / Occurrences.
* Activation. Roles / Types / Kinds.
* Alignment. Attributes / Values.
* Co relations (Algebras / CoSQL / NoSQL / LInQ):
* Models. Layers: Augmentation steps reification. Layers Statements Roles.
* Base Model:
* (Context, Occurrence / Statement, Attribute / Kind, Value / Entity);
* Discrete Layers Model: (Relationships / Assertions). Continuos CEP: 0 Duration Measures / Axis Events Relations.
* (Relationship, Relation, Kind, Entity);
* Context, Hierarchies, Roles, Causal, etc. Relationships / Relations.
* Continuous Layers Model: (Dimensions / Measures).  Discrete CEP: Order / Containment Relations.
* (Dimension, Measure, Unit, Value);
* Distance, Equivalence, Transportation, etc. Relations.
* Functional DCI: Monads / Functors / Data.
* Wrapper Types (Data Roles). Encoding.
* Wrapped types (Data Values). Encoding.
* Functors: Dataflow Domain / Range specifications (Contexts). Encoding.
* Monads: Dataflow implementations (Interactions). Encoding.
* Functional Roles. Wrapped Models Resources Monads Wrappers Encoding (Dataflow Entities):
* DCI Data (ETL / Ontology Mappings). Layers: Augmentation steps reification. Layer Statements Roles.
* Type: Class / Metaclass IDs
* Kind: Role / Occurrence IDs
* Node: Value / Instance IDs
* Arc: Attribute / Context IDs
* Arcs: (Instance, Occurrence / Node Class, Attribute, Value); Data.
* Nodes: (Class, Metaclass, Context, Role); Contexts.
* Mappings: Models. Interactions.
* Layers Hierarchy: Role : Class; Value : Instance. Property Graphs.
* Layers:
* Arcs: (Role, Instance, Metaclass, Context); DCI Interactions.
* (Context, Role, Instance, Class);
* (Class, Context, Role, Instance);
* Nodes: (Instance, Class, Context, Role); DCI Contexts.
* Layers:
* Arcs: (Arc, Node, Type, Kind); DCI Interactions.
* (Kind, Arc, Node, Type);
* (Type, Kind, Arc, Node);
* Nodes: (Node, Type, Kind, Arc); DCI Contexts.
* Mappings: Models. Interactions.
* Layers Hierarchy: Node : Arc;. Property Graphs.
* Models / Encoding:
* Layer Roles: (Context, Subject, Predicate, Object);
* Context: Kind Statements.
* Subject: Node Statements.
* Predicate: Arc Statements.
* Object: Node Statements.
* Type: Matching / Inferences.
* Encodings:
* Models: Quads / Property Graphs. DCI / Dataflow.
* Lists Model. Roles / Order.
* Hierarchical Graph Encoding.
* Sets.
* Functional DCI / Layers abstraction.
* Functional Parser. CUD. (data) of Grammar (Functional scheme) Dataflow Entities.
* Functional Parser. Grammar (Functional scheme) Dataflow Entities.
* Functional Parser. Parse (executions / behavior). Dataflow interactions instances.
* Codat: Dataflow / Protocol Prompts (run at).
* Protocol:
* Stateful I/O (ordered contexts). Reactive encoded Message driven gestures (CQRP).
* CDI / ESB Runtime. Backends. Connectors. CAM / CAN / DIDs.
* Dataflow HATEOAS.
* Augmentations / Alignments:
* Aggregation. Contexts / Occurrences.
* Activation. Roles / Types / Kinds.
* Alignment. Attributes / Values.