- ResourceURNs: Uniform identifiers across occurrences. DID URN. Endpoint. ResourceURN Statements: uniform functional metadata (contextual type / name, relations / aggregated occurrences). IDs Encodings.
- Templates / Monads / Sets Interfaces. Graph layout. Traversal (Quads Monads). Set Membership Function: Interfaces CSPOs Types Matching Signatures.
- Interface types differentiate in their CSPOs return value types (CSPOs type signatures: sets membership function).
-
- ResourceURN : (ResourceURN, Resource, Occurrence, Kind);
- ResourceURN aggregates Resource Occurrences Kind. Encodings.
- KindURN : (ResourceURN, Kind, Occurrence, Resource);
- ResourceURN aggregates Kind Occurrences Resources. Encodings.
- OccurrenceURN : (ResourceURN, Occurrence, Kind, Resource);
- ResourceURN aggregates Occurrences Kinds Resources. Encodings.
- SubjectResourceURN : (ResourceURN, Resource, Occurrence, SubjectKind);
-
- Functional APIs:
- Resource::getOccurrences
- Resource::getKinds
- Occurrence::getResources
- Occurrence::getKinds
- Kind::getResources
- Kind::getOccurrences
-
- Input / Canonical: Match Interfaces / Signatures: (Context, Occurrence, Attribute, Value);
- Attribute / Value Roles in matching interface context. Order / hierarchy encoding: functions (sorted wrapped functional collections: wrappers set comparators / aggregation in axis).
- Interface Quads Matching determine Sets (intersections) membership.
- Hashing: IResourceURN, IOccurrence, IKind, IResource. Nested recursive URNs aggregations. Aggregations / Order / Mappings / Traversal APIs.
-
- ResourceURN Occurrence, Kind, Resource Bindings Augmentation Service APIs:
-
- Naming Service:
- Input IRIs Encoding / Hashing. Input IRIs Matching. Endpoints (Messages Signatures). Semantic Hashing: DIDs. HATEOAS: Workflow states / referrers.
-
- Registry Service:
- ResourceURNs Statements bindings:
- Registry::resolve(ResourceURN) : dispatch to matching signatures:
- Registry::resolve(ResourceURN) : Resource
- Registry::resolve(ResourceURN) : Kind
- Registry::resolve(ResourceURN) : Occurrence
- Registry::resolveResourceURN(Resource, Occurrence, Kind) : Resource ResourceURN
- Registry::resolveResourceURN(Kind, Occurrence, Resource) : Kind ResourceURN
- Registry::resolveResourceURN(Occurrence, Kind, Resource) : Occurrence ResourceURN
-

- Index Service / Logs:
- Query Graphs of ResourceURN Nodes / Messages. Events driven Persistence.
-
- Functional Data Flow. ResourceURN Events. Message Logs Streams / Traversal (Index Persistence Events Graph Interfaces):
- (Occurrence, Kind, Resource)
- (Occurrence, Resource, Kind)
- (Kind, Occurrence, Resource)
- (Kind, Resource, Occurrence)
- (Resource, Occurrence, Kind)
- (Resource, Kind, Occurrence)
-
- Monads:
- ResourceURNMonad<ResourceClass : IResourceURN, etc.>
- ResourceMonad<ResourceClass : ISubjectResource, etc.>
- KindMonad<KindClass : ISubjectKind, etc.> Monad
- OccurrenceMonad<OccurrenceClass : ISubjectOccurrence, etc.> Monad
-
- Interfaces (Sets / CSPOs Roles). Kinds aggregate Resources, Resources aggregate Occurrences, Occurrences aggregate Kinds.
- IContext : measurement contexts. Statement (data / state), Mapping (schema), Transform (behavior) contexts.
- Reification: members of Kinds / Occurrences implements super sets types. Kinds of type implements that type. ToDo: resource or occurrence interfaces in statements signatures. Class patterns (multiple interfaces).
- Abstract interfaces: ISubject, etc. Align interfaces to CSPO roles (traversal / graph layout)
-
- IQuad : (IContext, ISubject, IPredicate, IObject)
- IResource : IQuad
- IKind : IQuad
- IOccurrence : IQuad
-
- Resource<Sets>
- Subject<Resource> : Resource<Subject> : Resource
- Predicate<Resource> : Resource
- Object<Resource> : Resource
- Kind<Sets> : Resource. Kind of Kind: SK(PK, OK). Contexts Kinds
- SubjectKind<Subject> : Kind
- PredicateKind<Predicate> : Kind
- ObjectKind<Object> : Kind
- Occurrence<Sets> : Kind
- Context<Subject> : Occurrence
- Context<Predicate> : Occurrence
- Context<Object> : Occurrence
-
- Kind<Kind<Subject<Context>>> : StatementKind
-
- Interfaces (Sets):

- IContext : ISubject, IPredicate, IObject
- ISubject : IResource
- IPredicate : IResource
- IObject : IResource
- ISubjectKind : IKind, IPredicateResource, IObjectResource
- IPredicateKind : IKind, ISubjectResource, IObjectResource
- IObjectKind : IKind, IPredicateResource, ISubjectResource
-
- Resources:
- ISubjectResource : IResource, ISubject
- (ISubjectKind, ISubjectOccurrence, IPredicateKind, IObjectKind)
- IPredicateResource : IResource, IPredicate
- (IPredicateKind, IPredicateOccurrence, ISubjectKind, IObjectKind)
- IObjectResource : IResource, IObject
- (IObjectKind, IObjectOccurrence, IPredicateKind, ISubjectKind)
- IStatementResource : IResource, IStatement
- (IStatementKind, IStatementOccurrence, IPredicateKind, IObjectKind)
- IMappingResource : IResource, IMapping
- (IMappingKind, IMappingOccurrence, ISubjectKind, IObjectKind)
- ITransformResource : IResource, ITransform
- (ITransformKind, ITransformOccurrence, IPredicateKind, ISubjectKind)
-
- Kinds:
- ISubjectKind
- (ISubjectOccurrence, ISubjectKind, IPredicateResource, IObjectResource)
- IPredicateKind
- (IPredicateOccurrence, IPredicateKind, ISubjectResource, IObjectResource)
- IObjectKind
- (IObjectOccurrence, IObjectKind, IPredicateResource, ISubjectResource)
- IStatementKind
- (IStatementOccurrence, SK of PK/OK: Relation, PK, OK)
- IMappingKind
- (IMappingOccurrence, PK of SK/OK: Schema, SK, OK)
- ITransformKind
- (ITransformOccurrence, OK of SK/PK: Behavior, SK, PK);
-
- Occurrences:
- ISubjectOccurrence : IOccurrence, ISubject
- (ISubjectResource, ISubjectKind, IPredicateOccurrence, IObjectOccurrence)
- IPredicateOccurrence : IOccurrence, IPredicate
- (IPredicateResource, IPredicateKind, ISubjectOccurrence, IObjectOccurrence)
- IObjectOccurrence : IOccurrence, IObject
- (IObjectResource, IObjectKind, ISubjectOccurrence, IPredicateOccurrence)
- IStatementOccurrence : IOccurrence, IStatement
- (IStatementResource, IStatementKind, IPredicateKind, IObjectKind)
- IMappingOccurrence : IOccurrence, IMapping
- (IMappingResource, IMappingKind, ISubjectKind, IObjectKind)
- ITransformOccurrence : IOccurrence, ITransform

- (ITransformResource, ITransformKind, IPredicateKind, ISubjectKind)
-
- Augmentations:
- Incremental / Feedback.
- Aggregations: ResourceURNs Source IRIs Sets / Layers streams / events (Resources, Occurrences, Kinds) parse / Occurrences population.
- Alignments: Aggregation traversal: ResourceURN URN IDs Model population. Merge / Matching, order / relations / contexts. Encoding (methods).
- Activations: Relationship Models I/O (DCI Layers / expanded SPO Aggregations feedback). DIDs URN hashing / generation (HATEOAS Endpoints). Data Flow.
-
- Encoding:
- IDs: Resource, Occurrence, Kind. ResourceURNs aggregation / order encoding.
- Graph / Tree List Parent / Child hierarchical encoding / hashing.
- Functional Data Flow. ResourceURN Events. Message Logs Streams / Traversal (Index Persistence Events Graph Interfaces):
- (Occurrence, Kind, Resource)
- (Occurrence, Resource, Kind)
- (Kind, Occurrence, Resource)
- (Kind, Resource, Occurrence)
- (Resource, Occurrence, Kind)
- (Resource, Kind, Occurrence)
- Quads / SPOs hierarchical list encoding.
- (C (S (P (O).
- Dataflow (value expressions). Signatures (events subscriptions: domain / range). Encode Order.
- Hashing: IResourceURN, IOccurrence, IKind, IResource. Nested recursive URNs aggregations. Order / Mappings / Traversal.
- Input / Canonical: Match Interfaces / Signatures: (Context, Occurrence, Attribute, Value); Attribute / Value Roles in matching interface context. Order / hierarchy encoding / functions (sorted wrapped functional collections: wrappers set comparators / aggregation axis).
-
- Order:
- StatementKind: PK(SK, OK). Relationship(Roles). Context, State, Mapping, Transform Kinds.
- SK(PK, OK)?
- OK(PK, SK)?
- Statement: abstract assertions (parsed / inferred). Mapping: abstract schema. Transform: abstract behavior. Relationships (dimensional / discrete): core model / ontology, Statement, Mapping, Transform synchronized (input / inferred Statement Events are fully parsed from CSPO Sets Layer).
- Order. States (Statements), Flows (Mappings), Events (Transforms). Kinds hierarchy tree / lattice (FCA). Action / Passion / State order. Kinds / Mappings domain / range Aggregation, Activation, Alignment. Comparisons. DCI / MVC / Relationships / Dimensional Aggregated upper onto matching gestures / flows.
- Input / Canonical: Match Interfaces / Signatures: (Context, Occurrence, Attribute, Value); Attribute / Value Roles in matching interface context. Order / hierarchy encoding /

functions (sorted wrapped functional collections: wrappers set comparators / aggregation axis).

-
- ResourceURNs DIDs:
- URN: DIDs. Endpoint APIs: Statements types / sets (Resource, Kind, Statement, Mapping, Transform OntResources hierarchy) content types / classes: Functional APIs. OntResource (DOM DTOs) quads representations references other DIDs, handle resolution, interactions, etc. via other DIDs endpoints and Resource Monad API.
- Method: did:ont:[ID]
- ID : OntClassName (Sets) ":" [HashedQuad];
- HashedQuad : [HashedURN] ":" [HashedURN] ":" [HashedURN] ":" [HashedURN];
- HashedURN : "[" HashedQuad "]" | HashedCSPOString;
- HashedCSPOString : Context ":" Subject ":" Predicate ":" Object;
- URN::ontResource (traversal parsed representation).
- OntResource::URN.
- Encoding: methods
- Hashing: four segments identifiers. Sets, binary octal digit order operable hashing (4 bit per segment). Aggregation: Statements graph layout. Occurrences. S-Expressions, MonParsec, CoSQL, map-reduce.
- URN: Encoded quad. Hashing: traversal, discovery, resolution. Merkle tree (DLT / Events). Encode typing / naming in context, about DID State Statements (hashing metadata):
- Occurrence ResourceURN : (ResourceURN, Occurrence, Kind, Resource);
- Kinds: Aggregate Attributes.
- State: Aggregate Kinds Resources Attributes / Values.
- Hierarchy: Kinds Attributes set (super) subset (sub) Kinds relationship.
- Order. Aggregation: Kinds / States lattice / tree. Populate / encode ResourceURNs order in contexts.
-
- DCI Context Model:
- DCI Relationship Contexts: Data / Schema / Behavior Model. DCI / MVC / Relationships Upper onto matching: gestures / flows.
- Metaclasses: PredicateKind SubjectKinds / ObjectKinds.
- Relationship: (Relationship, SubjectKind, PredicateKind, ObjectKind);
- PredicateKind of SK / OK. Employment(Employer, Employee); Employment (Employee, Position);
- Relation : (Relationship, Statements / Context, Role, Occurrence);
- Role : (Relation, Resource, Occurrence, Metaclass : Kinds);
- Occurrence : (Role, Relation, Context / Relation Statements, Resource);
-
- Discrete N-ary Relationship Aggregated Statements:
- Aggregated Statements traversal: expanded SPO form.
- Context: (Relationship : Predicate Kind, Relation : Statements, Role : Kind, Player : Resource);
- Predicate Kind of Reified S SK, O OK. (Relationship: Employment, Roles: Employee SK, Employer OK). Employment PK aggregated by Subjects and Objects Kinds. Relation Statements: Aggregated SK, PK, OK by Contexts Statement Kinds.
- (Working, workingRelationStmt, employer, IBM);

- (Working, workingRelationStmt, employee, John);
-
- Dimensional Relationships:
- (Dimension : Relationship, Measure : Relation, Unit : Kind, Value : Resource);
- (Time, oneHourStmt, minutes, 60);
- Distance Dimension: PK of Time SK / Meters OK. Define Dimension in terms of Relationship Kinds.
- Dimension Measure Statements: Domain PK Statements. Kind interface for Functional Transforms.
- Unit: PK Measure SK / OK Statement Kinds (SK / OK Members).
- Value: Dimension Measure Statement Kind Resource.
- Dimensional Order / Comparison: OrderKinds. Templates (populate).
- Order: Comparisons.
-
- Proof of Concept: Achieve REST Facade (synchronized) of Relationships given inputs from a system backend:
- Inputs: Aggregate SPO into CSPO: Aggregates Contexts Type / Table / Class Kinds. Aggregate PK Cols, Cols : Occurrence, Val : Resources.
- Inputs (Rel / Graph): (Type / Table / Class, PK : Resource, Col : Occurrence, Val : Resource).
- Inputs (Rel / Graph) FKs: Val : Resource equivalent PKs.
- Input / Canonical: Match Interfaces / Signatures: (Context, Occurrence, Attribute, Value);
-
- Encoding / Matching:
- Functional Context:
- Metaclass
- Class
- Instance
- Context
- Occurrence
- Role
- Hierarchy: Roles / Primitives (upper / aggregated hierarchy).
- Input / Canonical: Match Interfaces / Signatures: (Context, Occurrence, Attribute, Value); Attribute / Value Roles in matching interface context. Order / hierarchy encoding.
-
- Upper Ontology: Need, Product, Good, Purpose. Goal.
- Upper Ontologies: From Primitives to Forms / UI Gestures.
- Units of Measurement (continuos) APIs / Ontology.
- Discrete (events) APIs / Ontology. Relationships.
- Cube Statements:
- (Fact, Axis, Measure, Value);
-