Sets (Quads):

Sets (Resources, Subjects, Predicates, Objects, SubjectKinds, PredicateKinds, ObjectKinds, Statements: Mappings / Transforms) abstraction for representing Augmented RDF Graphs.

Augmentations:

Inference mechanisms for obtaining metadata from input statements and augmenting, for example, type information and models schema and transforms.

Data (Statement), Schema (Mapping), Behavior (Transform):

Aggregation

Activation

Alignment

Inputs / API:

I/O Normal Form: Statement
Service Facade. Functional Data Flow: Matching Mapping Transform: Statements. REST HATEOAS URNs:

I/O Statement:
(Context / Class, Instance, Attribute, Value);

Data Flow: Service Facade API:
REST Data Flow: Services Facade URN request / response HATEOAS flow.
Transform::Mapping::Statement::Kind::Resource;
Resource::Kind::Statement::Mapping::Transform;

Functional Sets Relations. Mappings / Transforms.

Sets (Wrappers): Parsed input Statements Resources instances are wrapped into Resource Monad Categories Statements:

Resource, CSPO Resources wrappers, Kinds Resources wrappers and Statement, Mapping, Transform wrappers.

Sets Resources REST HATEOAS / Data Flow IO Model Statements:

(Transform, Mapping, Statement, Kind);

Functional Data Flow:

Transform::Mapping::Statement::Kind::Resource;
Resource::Kind::Statement::Mapping::Transform;

Wrappers Sets:
CSPO Statement Form: for Transform Augmentations.
Functional Template Form: for aggregation inference mechanisms.

Wrapper Statements:
CSPO Statements Form:

Resources: Universe Set.
(Context : Resource, Occurrence : Resource, Attribute : Resource, Value : Resource); Normal Form.

Subjects:
(SubjectKind, Subject, Attribute : Predicate, Value : Object);

Predicates:
(PredicateKind, Attribute : Resource Subject, Predicate, Value : Object);

Objects: (ObjectKind, Attribute : Resource Predicate, Value : Resource Subject, Object);

SubjectKind (SK): Predicate / Object Intersection.
(Statement, SubjectKind, Attribute : Predicate, Value : Object);

PredicateKind (PK): Subject / Object intersection:
(Statement, Attribute : Subject, PredicateKind, Value : Object);

ObjectKind (OK): Predicate / Subject intersection. Occurring.
(Statement, Attribute : Predicate, Value : Subject, ObjectKind);

Statement / Mapping / Transform: Subject / Predicate / Object intersection:

Dual Occurrence of Transform (Mapping) / Occurring of Mapping (Transform) for Statements.
Templates: apply Transform Matching Mapping Statements.

Transforms:
(Context : Mapping, Occurrence : Resource, Attribute : Resource, Value : Resource);

Mappings:
(Context : Transform, Occurrence : Resource, Attribute : Resource, Value : Resource);

Statements:
(Context : Mapping, Occurrence : Resource, Attribute : Resource, Value : Resource);

Wrapper Statements:
Functional Template Form:

Resources: Universe Set.
(Context : Resource, Occurrence : Resource, Attribute : Resource, Value : Resource); Normal

Form.

Subjects:
(Subject, Occurrence : SubjectKind, Attribute : Resource P, Value : Resource O);

Predicates:
(Predicate, Occurrence: PredicateKind, Attribute : Resource S, Value : Resource O);

Objects: (Object, Occurrence : ObjectKind, Attribute : Resource P, Value : Resource S);

SubjectKind (SK): Predicate / Object Intersection. Occurrence:
(Context : SubjectKind, Occurrence : Statement, Attribute : Predicate, Value : Object);

PredicateKind (PK): Subject / Object intersection:
(Context : PredicateKind, Occurrence : Statement, Attribute : Subject, Value : Object);

ObjectKind (OK): Predicate / Subject intersection. Occurring.
(Context : ObjectKind, Occurrence : Statement, Attribute : Subject, Value : Predicate);

Statement / Mapping / Transform: Subject / Predicate / Object intersection:

Dual Occurrence of Transform (Mapping) / Occurring of Mapping (Transform) for Statements.
Templates: apply Transform Matching Mapping Statements.

Transforms:
(Context : Transform, Occurrence : Mapping, Attribute : Resource T, Value : Resource : U);

Mappings:
(Context : Mapping, Occurrence : Transform, Attribute : Resource T, Value : Resource U);

Statements :
(Context : Statement, Occurrence : Mapping, Attribute : Resource, Value : Resource);

Type Inference: Kinds (Classes):

Aggregate same Attributes occurrences for sets of Resources sharing same Attributes. Activate Context Transforms Kinds. Activate Kinds Resources Statements.

Wrapped Types (Kinds) Inputs Inference / Matching. Wrappers contains Wrapped CSPO Role Resources. Functional Flow into Occurrences, Attributes, Values.

Encodings. Representations: Instances / Literals Encoding. URNs. Resolution: sameAs Mappings / Parsing. Occurrence / Occurring domainOf / rangeOf Type Inference.

Model Kinds: Model Reified.
Domains Kinds: From inputs.
Reified Model Resource Kinds.
Functional: Monads (wrappers types / wrapped types inference). Kinds Domain Flow

(Mappings):

DOM Resources: dynamic object model / kinds.

Sample Statements:
(Dimension, Measure, Unit, Value);
(Relationship, Relation, Role, Player);
(Time, 1h, mins, 60m);
(Working, 1h, USD, 40);
(Working, 160h, USD, ?);

Augmentations:

Alignment:

Data: Statement; Matches Kinds / SPO: Mapping : Statements matching context.

Schema: Mapping; Matches Kinds / SPO: Transform : Mappings matching context.

Behavior: Transform; Matches Kinds / SPO: Transforms : Mapping Statements matching context.

Alignment API: Functional Functors / Transforms data flows.

Alignment:

Type Activation.

Aggregation:

Aggregation renders Resources into Functional Template Form (Statements / Mappings / Transforms updated) for Attribute / Value aggregations inference.

- Aggregate Resources for Attribute / Values. Statement: SPO Resource

- Aggregate SPO Resources:
- Subjects: (Subject, Occurrence : SubjectKind, Attribute : Resource P, Value : Resource O); Kinds

- Aggregate Kinds:
- (Context : SubjectKind, Occurrence : Statement, Attribute : Predicate, Value : Object); Statements

- Aggregate Statements:
- (Context : Statement, Occurrence : Mapping, Attribute : Resource, Value : Resource); Mappings

- Aggregate Mappings.

- (Context : Mapping: Occurrence : Transform, Attribute : Resource T, Value : Resource U); Transform

- Aggregate Transforms:
- (Context : Transform, Occurrence : Mapping, Attribute : Resource T, Value : Resource : U); Mapping

- Template Transforms (Mapping Roles): TO DO
- Aggregation: Statements for each Context Occurrence Attribute / Value. Mapping: for each matching Attribute / Value apply Transform, render Statement apply Attribute / Value Transform. Context: Transform / Class. Occurrence: Subject. Normal Form. Transform wrapped: Context / Class, Occurrence wrapped: Subject.
- Augmentations: Activation (Schema), Alignment (Data), Aggregation (Behavior) Matching (Mapping Function) results: Template Transforms (noop, merge, add); Transforms Flow State: listening for Matching Inputs.
- Aggregation: Statements for each Context Occurrence Attribute / Value. Mapping: for each matching Attribute / Value apply Transform, render Statement apply Attribute / Value Transform. Context: Transform / Class. Occurrence: Subject. Normal Form. Transform wrapped: Context / Class, Occurrence wrapped: Subject.
- Aggregation: Statements for each Context Occurrence Attribute / Value. Mapping: for each matching Attribute / Value apply Transform, render Statement apply Attribute / Value Transform. Context: Transform / Class. Occurrence: Subject. Normal Form. Transform wrapped: Context / Class, Occurrence wrapped: Subject.
- Alignment: Aggregate Resources Context Occurrences, Attributes, Values for Resource, Kinds, Statements, Mapings, Transforms Resources from Statement, Mapping, Transforms occurrences / occurring. Positional Roles: Functional Resources Roles Reification. Wrapper Types. Aggregation: Matching.
- Occurrence Object Member of Subject as Transform / Function Role.
- Occurring Subject Member of Object as Mapping / Function Role.
- Model Aggregation / Expansion (Augmentations match / apply) of Mappings / Transforms Core Statements
- Subscriptions: domain / range.
- Aggregation: Transforms Reified in Layers Contexts. Pattern Matching Template Layeresolved.ed
- (Wrapper, Wrapped, Mapping, Transform);
-
- Notes:
- URN : Resource (alignments). Primitives.
- Resource : Root Category. URN : Source / Surrogate Key / Crafted. Naming / Encodings (below).
- Ontology alignments: Data / Schema / Behavior Augmentations. Model / Schema / Upper / Domains: purposes / gestures (MVC / DCI Mappings / Transforms) layers. Example:
- Occurring / Context (Statements / Kinds)
- Roles (Metaclass, Class, Occurrence, Context, Role)
- MVC / DCI Mappings / Transforms. Example: Forms, Purpose, Gestures, Actors, Roles. Data / Schema / Behavior alignment.
- ESB: Endpoints, Features, Interfaces, Service Process Description / Discovery. Reactive Events Subscriptions. HATEOAS Endpoints "autowiring".

- BPM: Process, Steps, Flows, etc.
- Augmented Actionable (Process Flows, Items Activation) CMS. Browser: HATEOAS Protocol / APIs / Augmentations. Inferred / Reified / Resolvable Data Flows. Designer: Model Pallete. Declarative core / domains types / instances browsing / discovery "wiring".
- Graph Reified Grammars (upper). Terminal / Non Terminal. Rules / Productions. Mappings / Transform: browse grammar, rules, productions:
- (Rule, Context, lhs, rhs)
- Naming: Kinds / URNs Addressable Encodings. Parsing: URNs Encoded Functional Distributed Resource Resolution. Data Flow Transform / Mappings: Embedded Productions: Augmentations. NLP / NER. Ontology Matching: URN Class Transforms.
- Graph Embeddings: ML Backend Services (ML Predictions Augments Mappings / Transforms). Encodings (Naming).
- Encoding: Deep ML Embeddings. Data: classification, Schema: clustering, Behavior: regression.
- Naming: Auto Encoders. Semantic Hashing. Resources Mappings / Transforms Reified Maps / Tables. Keys / Values Resource Hashing / Resolution Functions: Contextual to Functional Environment State: Mappings Flows / Wrapped State.
- Naming: Augmentations. Contextual Hash Enabled: Functional Mapping Flows Map / Table Encoded / Resolved. Functional Relations: Ontology Matching / Aggregation / Inferences by Hash Encoded Metadata / Transforms Resolutions.
- Clients / Browsers: Peers. Protocol: Reactive Dialogs Prompts. Events. Distributed Data, Schema, Behavior Core Model Statements Encoded I/O: Layers Sync / Augmentation of Knowledge requested from each Peer(s) as Model inputs given resolution of Dialog (Subscriptions) event sourcing state. MVC / DCI Distributed State Transforms / Mappings. Augmented Peer(s) Models: updated View State (flows) / Mappings / Transforms. Rendezvous Peer Role. Local Peer: APIs for local / remote views (MVC / DCI) views (Web, REST) Rendering.
- (...)
-