Application:

Workflows (Domain Goals) general purpose ontology matching integration framework.

Reference Model:

Encodings:

Reference Model: Key / Value de-referenceable (for matching / embedding purposes) URIs having as host the peer that identified the Resource. DIDs resolution.

Messages:

(Context, Subject);

(Subject, Predicate);

(Predicate, Object);

Reference Model: Matching (recursive) of aggregated Rules (keys) Productions (values). Domains matches from upper to any level of (inter) Domain activations.

Layers: Productions in one layer are Rules of next layer (context, class, metaclass, instance, occurrence, role, etc.).

Messages flow from Domains layers matching Rules and producing further Productions (Messages) matching lower layers Contexts (Rules) until Service Resource (semiotic) Domain layer (Transform: URI service implementing interface). Transform semantics goes back until Domain layer: Augmentation of Resource layers (prompts / protocol semantics).

Domain Component Model:

Configures runtime (object / functional Domains) for Message parsing and behavior executions.

Domain Declarations: populate layers from Semiotic Context layer Domain description layers resources: Service Resources I/O layers matching / producing semiotic statements for Domain I/O.

Model: Triple Store. Meta Model Schema (RDF / RDFS). Upper Ontology. Primitives.

Controller: Resource Layers object hierarchy API. Named Transforms (Resource URI Service interface / implementation bindings) dataflow: signatures pipelines. Triple Store object graph (DTOs).

View: Layers Domain hierarchy functional API. Messages I/O. Protocol.

Bus. Messages / Layers I/O.

Triple Store:

Meta Model Hierarchy / Schema (Quad Layers / DOM):

CellValue : Value

ColumnField : Sign

ID : occurrence (PK) : Object

Context : instance (table) : Context

Role : metaclass (CSPO)

Resource : class. Monad Value (instance)

Kind : selector / transform (Functor mapping). Monad Value Type (metaclass / role)

Statement (context)

Relation : Kind Grammar (Productions). Monad Instance (occurrence)

Entity : Kind Grammar (Rules). Monad Type (class)

Relationship

Flow

Domain

Resource Layers object hierarchy API:

Functional Layers Domain model / transforms (events / controller). Named Transforms (Resource URI Service interface / implementation bindings) dataflow: signatures pipelines. Triple Store object graph (DTOs).

Value

Sign

Object

Context

Role

Resource

Kind

Statement

Relation

Entity

Relationship

Flow

Domain

TBD.

Layers Domain hierarchy functional API:

Monads AST / Parser Builder.

Layers Monads / Parser Monads (Messages : Rules / Productions). Functional events dataflow (selector signatures : Layer instance Activation).

DOM / AST Hierarchy:

Resource<OntResource[]>;

Kind<Resource[]>;

Statement<Kind[]>;

Relation<Statement[]>;

Entity<Relation[]>;

Relationship<Entity[]>;

Flow<Relationship<Entity[]>;

Domain<Flow[]>;

Bus:

Reference Model encoded Messages. Domains streams / pipelines. Messages enter Domains layer through Triple Store and returns back (results, prompts / populated). Messaging backend.

Component Domains:

Functional event driven Domains configures models behaviors. Domain types: service resources, interfaces (transforms), signatures (dataflow).

I/O / Persistence Domain.

Sets Augmentation Domain.

FCA AugmentationDomain.

Endpoints I/O Domain.

Dimensional Domain, Registry Domain, Index Domain, Naming Domain, DCI / MVC / Augmented / Declarative Forms / Flows Domain, etc. Business Domains. Other Domain types.

I/O / Persistence Domain:

Events (event sourcing). Backends. Peers. DIDs.

Sets Augmentation Domain:

TBD.

FCA Domain:

Semiotic mappings population. Augmentations: Aggregation (layers), Alignment (ontology), Activation (layers dataflows).

Endpoints Domain:

Forms / Flows (Grammar / Protocol Builder. Prompts). Resource augmentation endpoints. Forms / Flows browsing APIs.

Augmented Resources Contexts / Interactions Services.

OGM / Client Drivers Services.

Ontology Matching:

Upper Ontology. Grammars. Primitives.

Matching: Resource occurs as context / occurrence / atribute / value or class / occurrence / context / metaclass / instance in equivalent occurrence contexts (kinds / order / shapes / type hierarchies).

Meta Model encodes mappings for equivalence / relations hierarchies for entities instance occurrences in roles in contexts for concepts recursively till upper onto / primitive terms / relations.

Reify relation from / to predicates (semiotic) / relation entity (expanded relation entity / roles statements). Dimensional measures / state events. Shapes: transforms / rules.

FCA Ontology Matching: Upper ontology / primitives. Reference Model objects / attributes encoding. Encoding (scaling): lattice concepts relations / transforms traversal.

Semiotic mappings population. Augmentations: Aggregation (layers), Alignment (ontology), Activation (layers dataflow transforms: context products).

Reference Model Contexts.

Meta Model Layers Contexts.