Reactive Resources Stream Pipeline

Resource Monad : Resource<ResourceOccurrence>;

ContentType (Data / Transforms)

* onOccurrence transform
* getOccurrences(S, P, O) transform
* getOccurringContexts(S, P, O) transform
* fromRepresentation(Representation)
* toRepresentation(ContentType)
* ContentTypes? Encoding? Models types?

Representation : ContentType instance

* ContentType
* Encoded State

ResourceOccurrence

* Representation
* onOccurrence(ResourceOccurrence)
* getOccurrences(S, P, O)
* getOccurringContexts(S, P, O)
* getAttributes() : Attributes (by means of occurrences / schema)
  + getAttribute(Attribute)
  + setAttribute(Attribute, Value)

FCA contexts? Prime IDs? Sets contexts? Dimensional contexts? Activation contexts? Hierarchies?: ResourceOccurrence Models Schema?

ResourceOccurrence(s)

* IDOccurrence (URN / Prime ID Occurrence)
* SPO (Statement Occurrences)
  + Subject : IDOccurrence
  + Predicate : IDOccurrence
  + Object : IDOccurrence
* Kind<Player, Attribute, Value> (Role / Type. Statement Occurrences)
  + SubjectKind : Subject implements Kind<Subject, Predicate, Object>
  + PredicateKind : Predicate implements Kind<Predicate, Subject, Object>
  + ObjectKind : Object implements Kind<Object, Predicate, Subject>
* Statement (Data, Graph Occurrences)
* Statement (Kinds, Graph Occurrences)
  + Ex. (SK1, AttrX, ValY)
* Graph (Model Occurrences)
* Model

ResourceOccurrence(s) Activation (ContentType handled, Resource Monad bound):

Model::onOccurrence(Graph) : Statements.

Graph::onOccurrence(Statement) : IDOccurrence (SPOs).

Statement::onOccurrence(SPO) : ID.

SPO::onOccurrence(ID) : URN.

ID::onOccurrence(URN) : Prime ID.

getOccurrences(S, P, O)? (CPPE / RCV / FCA / Kinds / Alignment schema / instances inference. Filter / query / traversal).

Model::getOccurrences(S, P, O) : Graph.

Graph::getOccurrences(S, P, O) : Statement.

Statement::getOccurrences(S, P, O) : SPO.

SPO::getOccurrence(S, P, O) : ID.

ID::getOccurrences(S, P, O) : Prime ID

Events: Context::onOccurrence(Occurrence) : Result;

Main Event Loop:

Naming, Registry, Index stream pipeline steps.

Datasource input (URNs), Producer output (Graphs).

Input Events: (Context, Occurrence, Result) Occurrences.

Events Occurrences handler (topic):

onResource(Subject : Context, Predicate : Occurrence, Object : Result):

Resource Activation on each stream pipeline step (onOccurrence).

Main Event Loop Feedback:

Stream pipeline steps Events feed back for further augmentation. Step augmented Model in Events Context? getOccurrences?

* Models? Augmented Model in Events Context?
* Naming: Resource Factory. URN Crafting / Matching. Aggregation (type / state / order inference).
* In step context:
* Model::onOccurrence(Graph) : Statements.
* Graph::onOccurrence(Statement) : IDOccurrence (SPOs).
* Statement::onOccurrence(SPO) : ID.
* SPO::onOccurrence(ID) : URN.
* ID::onOccurrence(URN) : Prime ID.
* Registry: Resource Repository. URNs Resolution / CRUD. Alignment (equivalence / upper matching, link prediction).
* In step context:
* Model::onOccurrence(Graph) : Statements.
* Graph::onOccurrence(Statement) : IDOccurrence (SPOs).
* Statement::onOccurrence(SPO) : ID.
* SPO::onOccurrence(ID) : URN.
* ID::onOccurrence(URN) : Prime ID.
* Index: Resource Contents URNs Resolution (inferences, transforms). Activation (possible verbs / state changes / transforms).
* In step context:
* Model::onOccurrence(Graph) : Statements.
* Graph::onOccurrence(Statement) : IDOccurrence (SPOs).
* Statement::onOccurrence(SPO) : ID.
* SPO::onOccurrence(ID) : URN.
* ID::onOccurrence(URN) : Prime ID.