Vladimir Alexiev, PhD, PMP

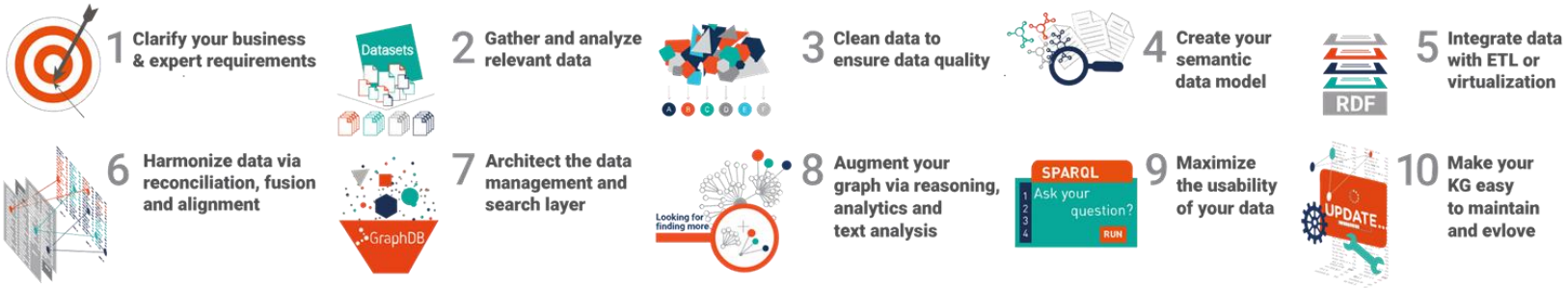
Ontotext Corp, Sofia, Bulgaria

Generation of Declarative Transformations from Semantic Models

Outline

- Building Knowledge Graphs in 10 steps
- Semantic Modeling vs Ontology Engineering; Polyglot Modeling
- **rdfpuml**: semantic models from turtle examples
- **rdf2sparql**: tabular transforms from turtle examples
- **rdf2rml**: R2RML transforms from turtle examples
- Backup slides: Semantic ETL desiderata
- Next time: tsv2owl, tsv2soml, owl2soml: ontologies and Semantic Object models from tabular data

Building Knowledge Graphs in 10 steps



- I'm Chief Data Architect at Ontotext, working at the Knowledge Graph Solutions group. We build KGs for a living
- Above is a simple 10-step recipe for KG building
  - Webinar: Knowledge Graphs: 5 Use Cases and 10 Steps to Get There
  - Video: Building Knowledge Graphs in 10 Steps
- This presentation is about steps 4 (models) and 5 (semantic ETL)

Semantic Modeling

Semantic Modeling vs Ontology Engineering

- Ontology engineering:
  - Know methodologies, modeling principles, top-level ontologies
  - Create ontologies mostly from scratch (green-field)
- Semantic modeling:
  - Research existing data standards and assets, harmonize them
  - Know and **reuse** existing ontologies, understand tradeoffs
  - Add classes and props when needed
  - Make holistic models that show how ontologies are used together
  - Document (human readable) and make machine readable artifacts (application profiles, RDF shapes, APIs, etc)
- For me semantic modeling is more than ontologies
  - Also, we like to do it in a very pragmatic way

Polyglot Modeling

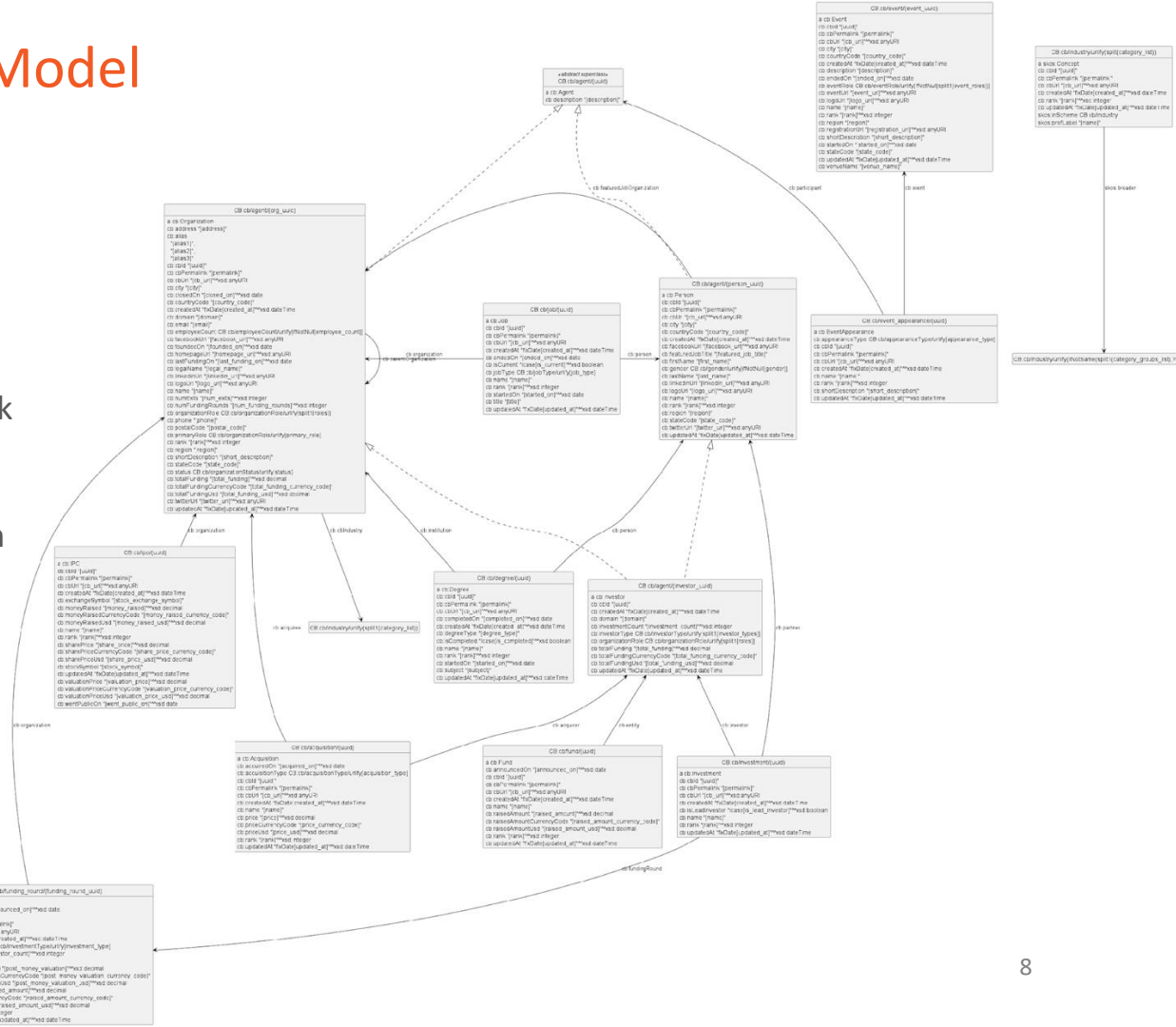
- Simple, technology-independent models
  - Understandable by domain experts: increases participation
- Generate various tech artefacts: semantic web doesn't (yet) rule the world!
  - Diagrams, documentation
  - RDFS, OWL, SHACL, SHEX, JSON-LD Context and Frame,
  - JSON schema, Elastic object model, etc, etc
  - "Write once. Use many. Creative laziness encouraged" [RAML]
- Examples:
  - FHIR: domain model to XML, JSON, JSONLD; XSD, JSON Schema, SHEX
  - LinkML: YAML-based models to various technical artefacts incl. JSON Schema, ontology, SHEX
  - Ontotext SOML: YAML-based model to SHACL and GraphQL (exposes KG parts to GraphQL)
  - Schema Salad (part of CWL): YAML-based model to JSON Schema and RDFS
  - A.ML and CloudInformationModel: YAML to RDFS, SHACL, SQL, R2RML
  - RAML (RESTful API Modeling Language): YAML to APIs (ala OpenAPI specifications)
- Links:
  - Many of these are in YAML: YAML-LD work group started (part of JSON-LD CG), yaml-ld#19 is about polyglot modeling
  - JSON-LD, YAML-LD and Polyglot Modeling. Alexiev, V. presentation, October 2022. slides link
  - Data Spaces vs Knowledge Graphs: How to Get To Semantic Data Spaces? Alexiev, V. In *Data Spaces & Semantic Interoperability Workshop*, Vienna, Austria, July 2022. Paper slides blog bibbase

Semantic Models from Turtle Examples

- https://github.com/VladimirAlexiev/rdf2rml: several open source tools. The first tool is:
- **rdfpuml**: generates model diagrams from examples in turtle. Lots of information and precision:
  - Ontologies, specific classes and properties
  - URL patterns
  - Datatypes and language tags
  - Simple data cleaning (e.g. SPLIT, URLIFY, FIX_DATE)
  - Use of source data (in parentheses)
  - Combining fields into the same property (e.g. alias1,2,3 mapped to multi-valued cb:alias)
  - Data of some nodes comes from multiple tables
- This is an **actionable model**, no hand-waving here
  - RDF by Example: rdfpuml for True RDF Diagrams, rdf2rml for R2RML Generation. Alexiev, V. Semantic Web in Libraries 2016 (SWIB 2016), Bonn, Germany, November 2016. slides html video bibbase

Crunchbase Semantic Model

- Crunchbase is a popular company dataset
- 18 CSV tables (with uuid and updated_at), 10M records, updated daily
- 2M companies, 929k detailed descriptions, 22k org parent links, 429k event appearances
- 1.5M people, 848k descriptions, 2.2M job positions, 495k degrees (education records)
- 223k investors, 488k funding rounds, 783k investments (participation in funding round), 137k acquisitions, 48k IPOs
- 750 industry classifications
- (Full image in slightly different style)



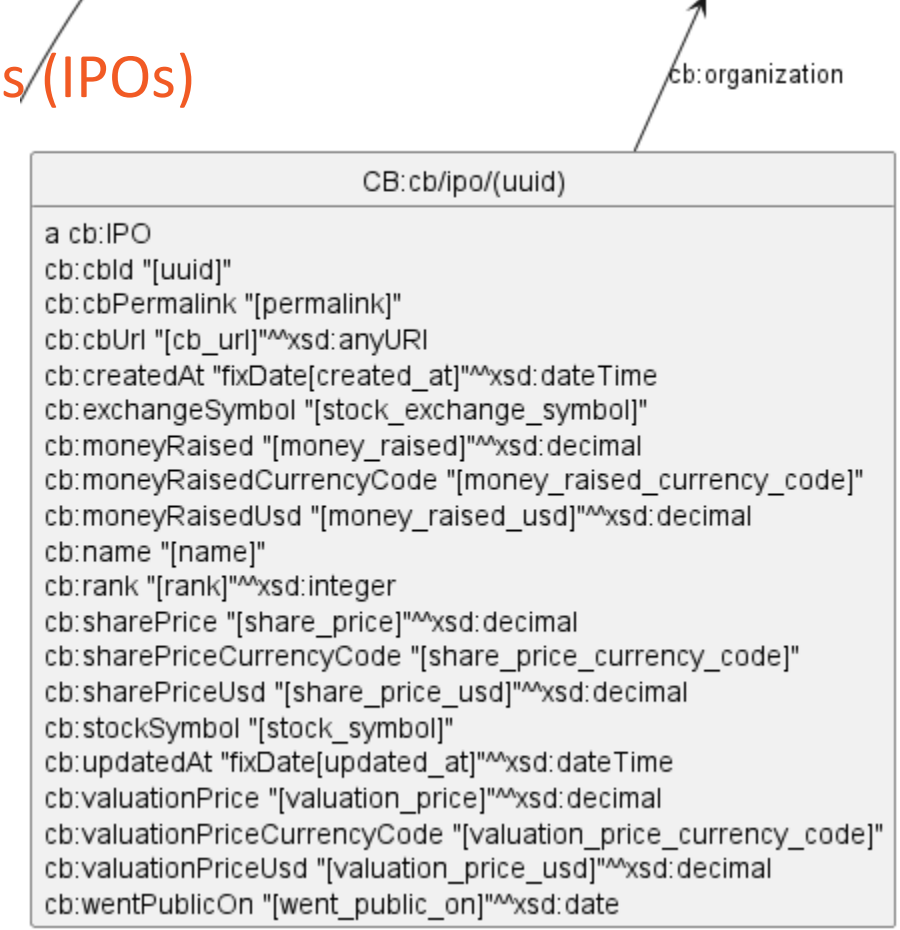
Focusing on Initial Public Offerings (IPOs)

Simple custom ontology:

- Exchange (exchangeSymbol) and ticker (stockSymbol) (e.g. NASDAQ:MSFT)
- Who (organization), when (wentPublicOn)
- Financial: moneyRaised, sharePrice, valuationPrice (in local currency and USD)
- cbPermalink, cbUrl (e.g. Microsoft IPO), CB rank
- Bookkeeping: uuid, createdAt, updatedAt (allow daily update)

Model elements:

- URL patterns
- Datatypes (e.g. **^^xsd:integer**)
- Simple fixes (e.g. **fixDate**)



Generating Tabular Transforms

Tabular Transforms From Turtle Examples

- **rdf2sparql**: generates SPARQL queries for transforming tabular data
- UPDATE for Ontotext Refine
  - ☺ Update directly to semantic repository: faster
  - ☺ Can join tabular to RDF data in a repository.
  - ☺ "Named-graph per table" for updates (locality and idempotence)
  - ☺ For CB we used "named graph per row" for fast daily update: GraphDB handles 10M named graphs just fine
- CONSTRUCT for Ontotext Refine.
  - ☹ Generates intermediate RDF file: slower
  - ☺ Can join tabular to RDF data in a repository
- CONSTRUCT queries for TARQL.
  - ☹ Generates intermediate RDF file: slower
  - ☹ Cannot join tabular to RDF data in a repository
- Handles datatypes, templated URLs, simple data cleaning (functions/macros)
- Chains functions applied to the same field (see example on the right)

- Turtle model using 2 functions:

```
cb:employeeCount <cb/employeeCount/urlify(ifNotNull(employee_count))>;
```

- Macros that define the functions:

```
#define urlify(x) bind(LCASE(REPLACE(REPLACE(REPLACE
  (x, "[^\\p{L}0-9]", "_"), "_+", "_"), "^_|_$", "")) as x##_URLIFY)
#define ifNotNull(x) bind(if(x in
  ("other","not provided","unknown"),?UNDEF,x) as x##_IFNOTNULL)
```

- SPARQL CONSTRUCT/UPDATE:

```
cb:employeeCount
  ?cb_employeeCount_employee_count_IFNOTNULL_URLIFY_URL;
```

- SPARQL WHERE binds to compute the variable:

```
bind(if(?employee_count in
  ("other","not provided","unknown"),?UNDEF, ?employee_count)
  as ?employee_count_IFNOTNULL)
bind(LCASE(REPLACE(REPLACE(REPLACE(?employee_count_IFNOTNULL,
   "[^\\p{L}0-9]", "_"), "_+", "_"), "^_|_$", ""))
  as ?employee_count_IFNOTNULL_URLIFY)
bind(iri(concat("cb/employeeCount/",
  ?employee_count_IFNOTNULL_URLIFY)) as
  ?cb_employeeCount_employee_count_IFNOTNULL_URLIFY_URL)
```

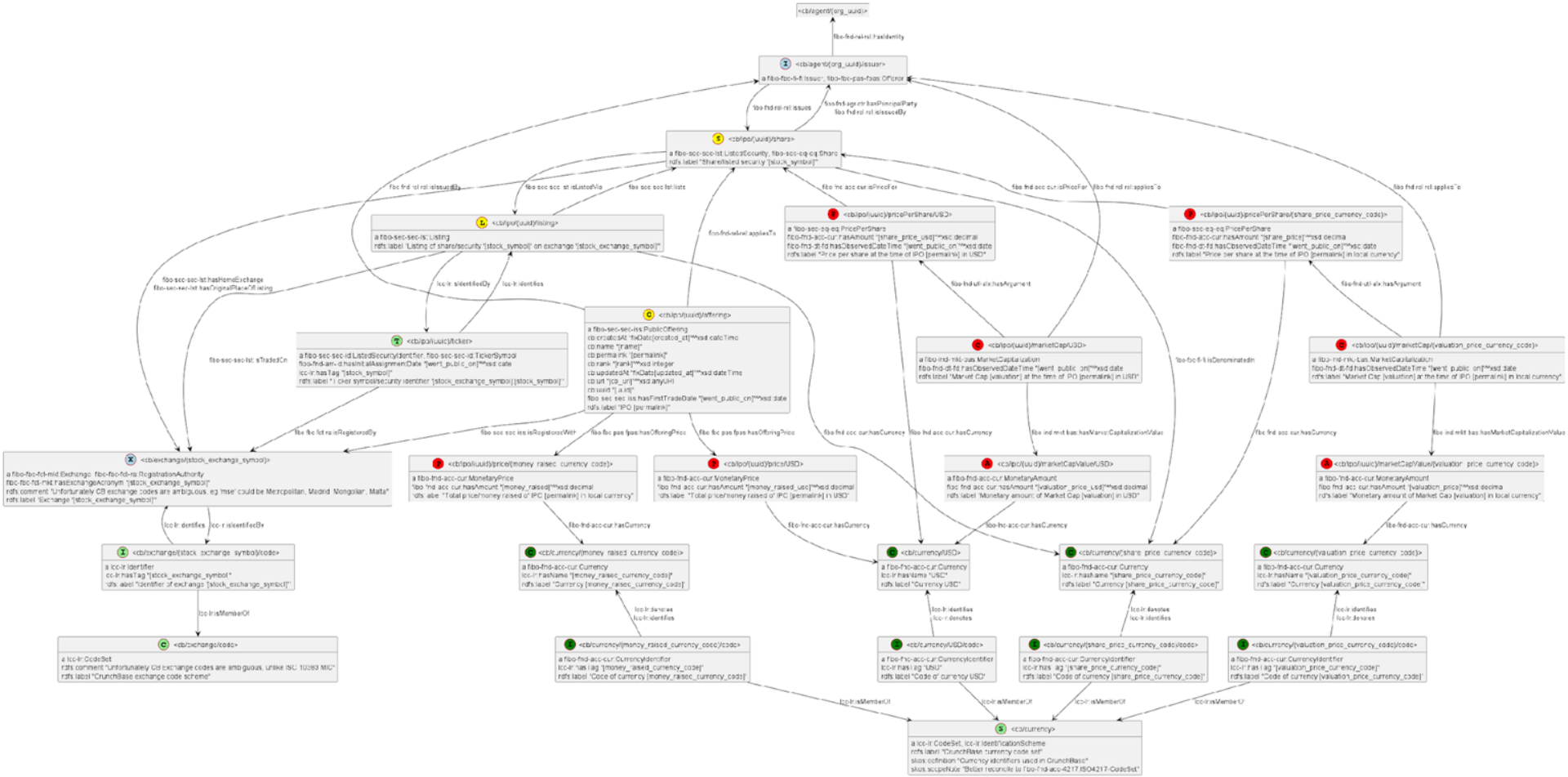
CB IPOs: From Model to SPARQL UPDATE

```
<cb/ipo/(uuid)> a cb:IPO;
  cb:cbId '(uuid)';
  cb:name '(name)';
  cb:cbPermalink '(permalink)';
  cb:cbUrl '(cb_url)'^^xsd:anyURI;
  cb:rank '(rank)'^^xsd:integer;
  cb:createdAt 'fixDate(created_at)'^^xsd:dateTime;
  cb:updatedAt 'fixDate(updated_at)'^^xsd:dateTime;
  cb:organization <cb/agent/(org_uuid)>;
  cb:exchangeSymbol '(stock_exchange_symbol)';
  cb:stockSymbol '(stock_symbol)';
  cb:wentPublicOn '(went_public_on)'^^xsd:date;
  cb:sharePriceUsd '(share_price_usd)'^^xsd:decimal;
  cb:sharePrice '(share_price)'^^xsd:decimal;
  cb:sharePriceCurrencyCode '(share_price_currency_code)';
  cb:valuationPriceUsd '(valuation_price_usd)'^^xsd:decimal;
  cb:valuationPrice '(valuation_price)'^^xsd:decimal;
  cb:valuationPriceCurrencyCode '(valuation_price_currency_code)';
  cb:moneyRaisedUsd '(money_raised_usd)'^^xsd:decimal;
  cb:moneyRaised '(money_raised)'^^xsd:decimal;
  cb:moneyRaisedCurrencyCode '(money_raised_currency_code)'.
```

```
delete {graph ?GRAPH {?s ?p ?o}}
where {
  service <rdf-mapper:ontorefine:PROJECT_ID> {
    bind(?c_updated_at as ?c_updated_at1)
    bind(iri(concat("cb/graph/ipos/",?c_uuid)) as ?GRAPH)}
  <cb> cb:updatedAt ?UPDATED_AT_DATETIME.
  bind(replace(str(?UPDATED_AT_DATETIME),'T',' ') as ?UPDATED_AT)
  filter(?c_updated_at1 > ?UPDATED_AT)
  graph ?GRAPH {?s ?p ?o}};
insert {graph ?GRAPH {
  ?cb_ipo_uuid_URL a cb:IPO;
    cb:cbId ?uuid;
    cb:name ?name;
    cb:cbPermalink ?permalink;
    cb:cbUrl ?CB_URL;
    cb:rank ?RANK;
    cb:createdAt ?CREATED_AT_FIXDATE;
    cb:updatedAt ?UPDATED_AT_FIXDATE;
    cb:organization ?cb_agent_org_uuid_URL;
    cb:exchangeSymbol ?stock_exchange_symbol;
    cb:stockSymbol ?stock_symbol;
    cb:wentPublicOn ?WENT_PUBLIC_ON;
    cb:sharePriceUsd ?SHARE_PRICE_USD;
    cb:sharePrice ?SHARE_PRICE;
    cb:sharePriceCurrencyCode ?share_price_currency_code;
    cb:valuationPriceUsd ?VALUATION_PRICE_USD;
    cb:valuationPrice ?VALUATION_PRICE;
    cb:valuationPriceCurrencyCode ?valuation_price_currency_code;
    cb:moneyRaisedUsd ?MONEY_RAISED_USD;
    cb:moneyRaised ?MONEY_RAISED;
    cb:moneyRaisedCurrencyCode ?money_raised_currency_code.
}}
where {
```

```
service <rdf-mapper:ontorefine:PROJECT_ID> {
  bind(?c_uuid as ?uuid)
  bind(?c_name as ?name)
  bind(?c_permalink as ?permalink)
  bind(?c_cb_url as ?cb_url)
  bind(?c_rank as ?rank)
  bind(?c_created_at as ?created_at)
  bind(?c_updated_at as ?updated_at)
  bind(?c_org_uuid as ?org_uuid)
  bind(?c_stock_exchange_symbol as ?stock_exchange_symbol)
  bind(?c_stock_symbol as ?stock_symbol)
  bind(?c_went_public_on as ?went_public_on)
  bind(?c_share_price_usd as ?share_price_usd)
  bind(?c_share_price as ?share_price)
  bind(?c_share_price_currency_code as ?share_price_currency_code)
  bind(?c_valuation_price_usd as ?valuation_price_usd)
  bind(?c_valuation_price as ?valuation_price)
  bind(?c_valuation_price_currency_code as ?valuation_price_currency_code)
  bind(?c_money_raised_usd as ?money_raised_usd)
  bind(?c_money_raised as ?money_raised)
  bind(?c_money_raised_currency_code as ?money_raised_currency_code)
  bind(iri(concat("cb/ipo/",?uuid)) as ?cb_ipo_uuid_URL)
  bind(strdt(?cb_url,xsd:anyURI) as ?CB_URL)
  bind(strdt(?rank,xsd:integer) as ?RANK)
  bind(REPLACE(?created_at,' ','T') as ?created_at_FIXDATE)
  bind(strdt(?created_at_FIXDATE,xsd:dateTime) as ?CREATED_AT_FIXDATE)
  bind(REPLACE(?updated_at,' ','T') as ?updated_at_FIXDATE)
  bind(strdt(?updated_at_FIXDATE,xsd:dateTime) as ?UPDATED_AT_FIXDATE)
  bind(iri(concat("cb/agent/",?org_uuid)) as ?cb_agent_org_uuid_URL)
  bind(strdt(?went_public_on,xsd:date) as ?WENT_PUBLIC_ON)
  bind(strdt(?share_price_usd,xsd:decimal) as ?SHARE_PRICE_USD)
  bind(strdt(?share_price,xsd:decimal) as ?SHARE_PRICE)
  bind(strdt(?valuation_price_usd,xsd:decimal) as ?VALUATION_PRICE_USD)
  bind(strdt(?valuation_price,xsd:decimal) as ?VALUATION_PRICE)
  bind(strdt(?money_raised_usd,xsd:decimal) as ?MONEY_RAISED_USD)
  bind(strdt(?money_raised,xsd:decimal) as ?MONEY_RAISED)
  bind(?c_updated_at as ?c_updated_at1)
  bind(iri(concat("cb/graph/ipos/",?c_uuid)) as ?GRAPH)}
<cb> cb:updatedAt ?UPDATED_AT_DATETIME.
bind(replace(str(?UPDATED_AT_DATETIME),'T',' ') as ?UPDATED_AT)
filter(?c_updated_at1 > ?UPDATED_AT)}
```

Could Go a Lot More Complex: IPOs in FIBO



IPOs in FIBO: Turtle model (165 lines)

```
# GRAPH <cb/ipos>

## Issuer as role of a CB agent

<cb/agent/(org_uuid)/issuer> a fibo-fbc-fi-fi:Issuer, fibo-fbc-pas-fpas:Of
  fibo-fnd-rel-rel:issues <cb/ipo/(uuid)/share>;
  fibo-fnd-rel-rel:hasIdentity <cb/agent/(org_uuid)>.

## Stock exchange

<cb/exchange/(stock_exchange_symbol)> a fibo-fbc-fct-mkt:Exchange, fibo-fbc
  rdfs:label "Exchange '(stock_exchange_symbol)'";
  rdfs:comment "Unfortunately CB exchange codes are ambiguous, eg 'mse' co
  fibo-fbc-fct-mkt:hasExchangeAcronym '(stock_exchange_symbol)';
  lcc-lr:isIdentifiedBy <cb/exchange/(stock_exchange_symbol)/code>.

<cb/exchange/(stock_exchange_symbol)/code> a lcc-lr:Identifier;
  rdfs:label "Identifier of exchange '(stock_exchange_symbol)'";
  lcc-lr:hasTag       '(stock_exchange_symbol)';
  lcc-lr:identifies   <cb/exchange/(stock_exchange_symbol)>;
  lcc-lr:isMemberOf   <cb/exchange/code>.

<cb/exchange/code> a lcc-lr:CodeSet;
  rdfs:label 'CrunchBase exchange code scheme';
  rdfs:comment 'Unfortunately CB Exchange codes are ambiguous, unlike ISO

## Offering, Share, Listing, Ticker

<cb/ipo/(uuid)/offering> a fibo-sec-sec-iss:PublicOffering;
  rdfs:label "IPO (permalink)";
  fibo-fbc-pas-fpas:hasOfferingPrice <cb/ipo/(uuid)/price/(money_raised_cur
  fibo-fnd-rel-rel:appliesTo <cb/ipo/(uuid)/share>;
  fibo-fnd-rel-rel:isIssuedBy <cb/agent/(org_uuid)/issuer>;
  fibo-sec-sec-iss:hasFirstTradeDate '(went_public_on)'^^xsd:date;
  fibo-sec-sec-iss:isRegisteredWith <cb/exchange/(stock_exchange_symbol)>;
  cb:uuid '(uuid)';
  cb:name '(name)';
  cb:permalink '(permalink)';
  cb:url '(cb_url)'^^xsd:anyURI;
  cb:rank '(rank)'^^xsd:integer;
  cb:createdAt 'fixDate(created_at)'^^xsd:dateTime;
  cb:updatedAt 'fixDate(updated_at)'^^xsd:dateTime.

<cb/ipo/(uuid)/share> a fibo-sec-sec-lst:ListedSecurity, fibo-sec-eq-eq:Sh
  rdfs:label "Share/listed security '(stock_symbol)'";
  fibo-sec-sec-lst:isListedVia <cb/ipo/(uuid)/listing>;
  fibo-fnd-agr-ctr:hasPrincipalParty <cb/agent/(org_uuid)/issuer>;
  fibo-fnd-rel-rel:isIssuedBy <cb/agent/(org_uuid)/issuer>;
  fibo-fbc-fi-fi:isDenominatedIn <cb/currency/(share_price_currency_code)>
  fibo-sec-sec-lst:hasOriginalPlaceOfListing <cb/exchange/(stock_exchange_
  fibo-sec-sec-lst:hasHomeExchange <cb/exchange/(stock_exchange_symbol)>.

<cb/ipo/(uuid)/listing> a fibo-sec-sec-lst:Listing;
  rdfs:label "Listing of share/security '(stock_symbol)' on exchange '(sto
  fibo-sec-sec-lst:lists <cb/ipo/(uuid)/share>;
  lcc-lr:isIdentifiedBy <cb/ipo/(uuid)/ticker>;
  fibo-sec-sec-lst:isTradedOn <cb/exchange/(stock_exchange_symbol)>;
  fibo-fnd-acc-cur:hasCurrency <cb/currency/(share_price_currency_code)>.

<cb/ipo/(uuid)/ticker> a fibo-sec-sec-id:TickerSymbol, fibo-sec-sec-id:Lis
  rdfs:label "Ticker symbol/security identifier '(stock_exchange_symbol):(
  lcc-lr:hasTag '(stock_symbol)';
  fibo-fbc-fct-ra:isRegisteredBy <cb/exchange/(stock_exchange_symbol)>;
  lcc-lr:identifies <cb/ipo/(uuid)/listing>;
  fibo-fnd-arr-id:hasInitialAssignmentDate '(went_public_on)'^^xsd:date.

## Financials

<cb/ipo/(uuid)/price/(money_raised_currency_code)> a fibo-fnd-acc-cur:Mone
  rdfs:label "Total price/money raised of IPO (permalink) in local currenc
  fibo-fnd-acc-cur:hasAmount '(money_raised)'^^xsd:decimal;
  fibo-fnd-acc-cur:hasCurrency <cb/currency/(money_raised_currency_code)>.

<cb/ipo/(uuid)/price/USD> a fibo-fnd-acc-cur:MonetaryPrice;
  rdfs:label "Total price/money raised of IPO (permalink) in USD";
  fibo-fnd-acc-cur:hasAmount '(money_raised_usd)'^^xsd:decimal;
  fibo-fnd-acc-cur:hasCurrency <cb/currency/USD>.

<cb/ipo/(uuid)/marketCap/(valuation_price_currency_code)> a fibo-ind-mkt-b
  rdfs:label "Market Cap (valuation) at the time of IPO (permalink) in loc
  fibo-ind-mkt-bas:hasMarketCapitalizationValue <cb/ipo/(uuid)/marketCapVa
  fibo-fnd-utl-alx:hasArgument <cb/ipo/(uuid)/pricePerShare/(share_price_c
  fibo-fnd-dt-fd:hasObservedDateTime '(went_public_on)'^^xsd:date;
  fibo-fnd-rel-rel:appliesTo <cb/agent/(org_uuid)/issuer>.

<cb/ipo/(uuid)/marketCap/USD> a fibo-ind-mkt-bas:MarketCapitalization;
  rdfs:label "Market Cap (valuation) at the time of IPO (permalink) in USD
  fibo-ind-mkt-bas:hasMarketCapitalizationValue <cb/ipo/(uuid)/marketCapVa
  fibo-fnd-utl-alx:hasArgument <cb/ipo/(uuid)/pricePerShare/USD>;
  fibo-fnd-dt-fd:hasObservedDateTime '(went_public_on)'^^xsd:date;
  fibo-fnd-rel-rel:appliesTo <cb/agent/(org_uuid)/issuer>.

<cb/ipo/(uuid)/marketCapValue/(valuation_price_currency_code)> a fibo-fnd-
  rdfs:label "Monetary amount of Market Cap (valuation) in local currency"
  fibo-fnd-acc-cur:hasAmount '(valuation_price)'^^xsd:decimal;
  fibo-fnd-acc-cur:hasCurrency <cb/currency/(valuation_price_currency_code

<cb/ipo/(uuid)/marketCapValue/USD> a fibo-fnd-acc-cur:MonetaryAmount;
  rdfs:label "Monetary amount of Market Cap (valuation) in USD";
  fibo-fnd-acc-cur:hasAmount '(valuation_price_usd)'^^xsd:decimal;
  fibo-fnd-acc-cur:hasCurrency <cb/currency/USD>.

<cb/ipo/(uuid)/pricePerShare/(share_price_currency_code)> a fibo-sec-eq-eq
  rdfs:label "Price per share at the time of IPO (permalink) in local curr
  fibo-fnd-acc-cur:hasAmount '(share_price)'^^xsd:decimal;
  fibo-fnd-acc-cur:hasCurrency <cb/currency/(share_price_currency_code)>;
  fibo-fnd-dt-fd:hasObservedDateTime '(went_public_on)'^^xsd:date;
  fibo-fnd-acc-cur:isPriceFor <cb/ipo/(uuid)/share>.

<cb/ipo/(uuid)/pricePerShare/USD> a fibo-sec-eq-eq:PricePerShare;
  rdfs:label "Price per share at the time of IPO (permalink) in USD";
  fibo-fnd-acc-cur:hasAmount '(share_price_usd)'^^xsd:decimal;
  fibo-fnd-acc-cur:hasCurrency <cb/currency/USD>;
  fibo-fnd-dt-fd:hasObservedDateTime '(went_public_on)'^^xsd:date;
  fibo-fnd-acc-cur:isPriceFor <cb/ipo/(uuid)/share>.


## Currencies: USD plus 3 more for the 3 financials

<cb/currency/USD> a fibo-fnd-acc-cur:Currency;
  lcc-lr:hasName 'USD';
  rdfs:label 'Currency USD'.
<cb/currency/USD/code> a fibo-fnd-acc-cur:CurrencyIdentifier;
  rdfs:label 'Code of currency USD';
  lcc-lr:hasTag 'USD';
  lcc-lr:denotes <cb/currency/USD>;
  lcc-lr:identifies <cb/currency/USD>;
  lcc-lr:isMemberOf <cb/currency>.

<cb/currency/(share_price_currency_code)> a fibo-fnd-acc-cur:Currency;
  lcc-lr:hasName '(share_price_currency_code)';
  rdfs:label 'Currency (share_price_currency_code)'.
<cb/currency/(share_price_currency_code)/code> a fibo-fnd-acc-cur:Currency
  rdfs:label 'Code of currency (share_price_currency_code)';
  lcc-lr:hasTag '(share_price_currency_code)';
  lcc-lr:denotes <cb/currency/(share_price_currency_code)>;
  lcc-lr:identifies <cb/currency/(share_price_currency_code)>;
  lcc-lr:isMemberOf <cb/currency>.

<cb/currency/(valuation_price_currency_code)> a fibo-fnd-acc-cur:Currency;
  lcc-lr:hasName '(valuation_price_currency_code)';
  rdfs:label 'Currency (valuation_price_currency_code)'.
<cb/currency/(valuation_price_currency_code)/code> a fibo-fnd-acc-cur:Curr
  rdfs:label 'Code of currency (valuation_price_currency_code)';
  lcc-lr:hasTag '(valuation_price_currency_code)';
  lcc-lr:denotes <cb/currency/(valuation_price_currency_code)>;
  lcc-lr:identifies <cb/currency/(valuation_price_currency_code)>;
  lcc-lr:isMemberOf <cb/currency>.

<cb/currency/(money_raised_currency_code)> a fibo-fnd-acc-cur:Currency;
  lcc-lr:hasName '(money_raised_currency_code)';
  rdfs:label 'Currency (money_raised_currency_code)'.
<cb/currency/(money_raised_currency_code)/code> a fibo-fnd-acc-cur:Currenc
  rdfs:label 'Code of currency (money_raised_currency_code)';
  lcc-lr:hasTag '(money_raised_currency_code)';
  lcc-lr:denotes <cb/currency/(money_raised_currency_code)>;
  lcc-lr:identifies <cb/currency/(money_raised_currency_code)>;
  lcc-lr:isMemberOf <cb/currency>.

<cb/currency> a lcc-lr:IdentificationScheme, lcc-lr:CodeSet;
  rdfs:label 'CrunchBase currency code set';
  skos:definition 'Currency identifiers used in CrunchBase';
  skos:scopeNote 'Better reconcile to fibo-fnd-acc-4217:ISO4217-CodeSet'.
```

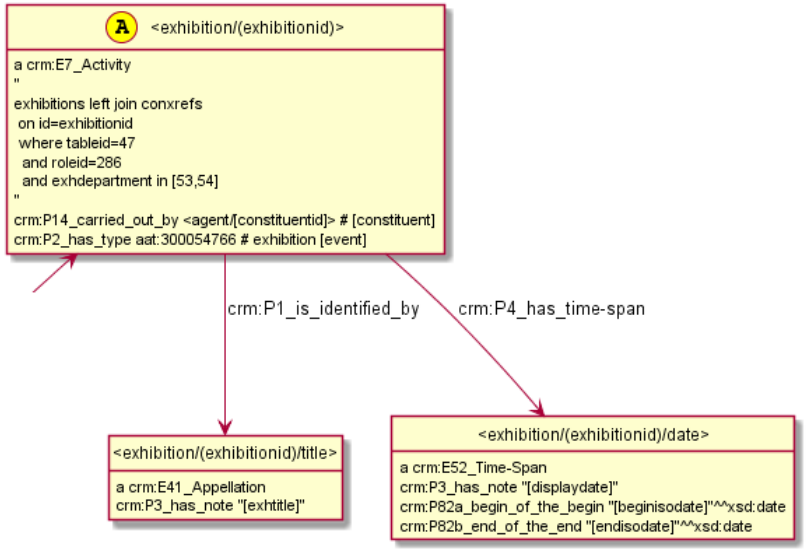
IPOs in FIBO: SPARQL (220 lines, 56 binds, 21 prefixes)

```
base <https://ontotext.com/crunchbase/resource/>
prefix cb: <https://ontotext.com/crunchbase/ontology/>
prefix fibo-fbc-fct-mkt: <https://spec.edmcouncil.org/fibo/ontology/FBC/FunctionalEnt
prefix fibo-fbc-fct-ra: <https://spec.edmcouncil.org/fibo/ontology/FBC/FunctionalEnti
prefix fibo-fbc-fi-fi: <https://spec.edmcouncil.org/fibo/ontology/FBC/FinancialInstru
prefix fibo-fbc-pas-fpas: <https://spec.edmcouncil.org/fibo/ontology/FBC/ProductsAndS
prefix fibo-fnd-acc-cur: <https://spec.edmcouncil.org/fibo/ontology/FND/Accounting/Cu
prefix fibo-fnd-agr-ctr: <https://spec.edmcouncil.org/fibo/ontology/FND/Agreements/Co
prefix fibo-fnd-arr-id: <https://spec.edmcouncil.org/fibo/ontology/FND/Arrangements/I
prefix fibo-fnd-dt-fd: <https://spec.edmcouncil.org/fibo/ontology/FND/DatesAndTimes/F
prefix fibo-fnd-rel-rel: <https://spec.edmcouncil.org/fibo/ontology/FND/Relations/Rel
prefix fibo-fnd-utl-alx: <https://spec.edmcouncil.org/fibo/ontology/FND/Utilities/Ana
prefix fibo-ind-mkt-bas: <https://spec.edmcouncil.org/fibo/ontology/IND/MarketIndices
prefix fibo-sec-eq-eq: <https://spec.edmcouncil.org/fibo/ontology/SEC/Equities/Equity
prefix fibo-sec-sec-id: <https://spec.edmcouncil.org/fibo/ontology/SEC/Securities/Sec
prefix fibo-sec-sec-iss: <https://spec.edmcouncil.org/fibo/ontology/SEC/Securities/Se
prefix fibo-sec-sec-lst: <https://spec.edmcouncil.org/fibo/ontology/SEC/Securities/Se
prefix lcc-lr: <https://www.omg.org/spec/LCC/Languages/LanguageRepresentation/>
prefix puml: <http://plantuml.com/ontology#>
prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#>
prefix skos: <http://www.w3.org/2004/02/skos/core#>
prefix xsd: <http://www.w3.org/2001/XMLSchema#>
clear silent graph <cb/ipos>;
insert {graph <cb/ipos> {
  ## Issuer as role of a CB agent
  ?cb_agent_org_uuid_issuer_URL a fibo-fbc-fi-fi:Issuer, fibo-fbc-pas-fpas:Offeror;
    fibo-fnd-rel-rel:issues ?cb_ipo_uuid_share_URL;
    fibo-fnd-rel-rel:hasIdentity ?cb_agent_org_uuid_URL.
  ## Stock exchange
  ?cb_exchange_stock_exchange_symbol_URL a fibo-fbc-fct-mkt:Exchange, fibo-fbc-fct-ra
    rdfs:label "Exchange ?stock_exchange_symbol";
    rdfs:comment "Unfortunately CB exchange codes are ambiguous, eg 'mse' could be Me
    fibo-fbc-fct-mkt:hasExchangeAcronym ?stock_exchange_symbol;
    lcc-lr:isIdentifiedBy ?cb_exchange_stock_exchange_symbol_code_URL.
  ?cb_exchange_stock_exchange_symbol_code_URL a lcc-lr:Identifier;
    rdfs:label "Identifier of exchange ?stock_exchange_symbol";
    lcc-lr:hasTag ?stock_exchange_symbol;
    lcc-lr:identifies ?cb_exchange_stock_exchange_symbol_URL;
    lcc-lr:isMemberOf <cb/exchange/code>.
  <cb/exchange/code> a lcc-lr:CodeSet;
    rdfs:label 'CrunchBase exchange code scheme';
    rdfs:comment 'Unfortunately CB Exchange codes are ambiguous, unlike ISO 10383 MIC
  ## Offering, Share, Listing, Ticker
  ?cb_ipo_uuid_offering_URL a fibo-sec-sec-iss:PublicOffering;
    rdfs:label "IPO (permalink)";
    fibo-fbc-pas-fpas:hasOfferingPrice ?cb_ipo_uuid_price_money_raised_currency_code_
    fibo-fnd-rel-rel:appliesTo ?cb_ipo_uuid_share_URL;
    fibo-fnd-rel-rel:isIssuedBy ?cb_agent_org_uuid_issuer_URL;
    fibo-sec-sec-iss:hasFirstTradeDate ?went_public_on_xsd_date;
    fibo-sec-sec-iss:isRegisteredWith ?cb_exchange_stock_exchange_symbol_URL;
    cb:uuid ?uuid;
    cb:name ?name;
    cb:permalink ?permalink;
    cb:url ?cb_url_xsd_anyURI;
    cb:rank ?rank_xsd_integer;
    cb:createdAt ?created_at_FIXDATE_xsd_dateTime;
    cb:updatedAt ?updated_at_FIXDATE_xsd_dateTime.
  ?cb_ipo_uuid_share_URL a fibo-sec-sec-lst:ListedSecurity, fibo-sec-eq-eq:Share;
    rdfs:label "Share/listed security ?stock_symbol";
    fibo-sec-sec-lst:isListedVia ?cb_ipo_uuid_listing_URL;
    fibo-fnd-agr-ctr:hasPrincipalParty ?cb_agent_org_uuid_issuer_URL;
    fibo-fnd-rel-rel:isIssuedBy ?cb_agent_org_uuid_issuer_URL;
    fibo-fbc-fi-fi:isDenominatedIn ?cb_currency_share_price_currency_code_URL;
    fibo-sec-sec-lst:hasOriginalPlaceOfListing ?cb_exchange_stock_exchange_symbol_URL
    fibo-sec-sec-lst:hasHomeExchange ?cb_exchange_stock_exchange_symbol_URL.
  ?cb_ipo_uuid_listing_URL a fibo-sec-sec-lst:Listing;
    rdfs:label "Listing of share/security ?stock_symbol on exchange ?stock_exchange_s
    fibo-sec-sec-lst:lists ?cb_ipo_uuid_share_URL;
    lcc-lr:isIdentifiedBy ?cb_ipo_uuid_ticker_URL;
    fibo-sec-sec-lst:isTradedOn ?cb_exchange_stock_exchange_symbol_URL;
    fibo-fnd-acc-cur:hasCurrency ?cb_currency_share_price_currency_code_URL.
  ?cb_ipo_uuid_ticker_URL a fibo-sec-sec-id:TickerSymbol, fibo-sec-sec-id:ListedSecur
    rdfs:label "Ticker symbol/security identifier '(stock_exchange_symbol):(stock_sym
    lcc-lr:hasTag ?stock_symbol;
    fibo-fbc-fct-ra:isRegisteredBy ?cb_exchange_stock_exchange_symbol_URL;
    lcc-lr:identifies ?cb_ipo_uuid_listing_URL;
    fibo-fnd-arr-id:hasInitialAssignmentDate ?went_public_on_xsd_date.
  ## Financials
  ?cb_ipo_uuid_price_money_raised_currency_code_URL a fibo-fnd-acc-cur:MonetaryPrice;
    rdfs:label "Total price/money raised of IPO (permalink) in local currency";
    fibo-fnd-acc-cur:hasAmount ?money_raised_xsd_decimal;
    fibo-fnd-acc-cur:hasCurrency ?cb_currency_money_raised_currency_code_URL.
  ?cb_ipo_uuid_price_USD_URL a fibo-fnd-acc-cur:MonetaryPrice;
    rdfs:label "Total price/money raised of IPO (permalink) in USD";
    fibo-fnd-acc-cur:hasAmount ?money_raised_usd_xsd_decimal;
    fibo-fnd-acc-cur:hasCurrency <cb/currency/USD>.
  ?cb_ipo_uuid_marketCap_valuation_price_currency_code_URL a fibo-ind-mkt-bas:MarketC
    rdfs:label "Market Cap (valuation) at the time of IPO (permalink) in local curren
    fibo-ind-mkt-bas:hasMarketCapitalizationValue ?cb_ipo_uuid_marketCapValue_valuati
    fibo-fnd-utl-alx:hasArgument ?cb_ipo_uuid_pricePerShare_share_price_currency_code
    fibo-fnd-dt-fd:hasObservedDateTime ?went_public_on_xsd_date;
    fibo-fnd-rel-rel:appliesTo ?cb_agent_org_uuid_issuer_URL.
  ?cb_ipo_uuid_marketCap_USD_URL a fibo-ind-mkt-bas:MarketCapitalization;
    rdfs:label "Market Cap (valuation) at the time of IPO (permalink) in USD";
    fibo-ind-mkt-bas:hasMarketCapitalizationValue ?cb_ipo_uuid_marketCapValue_USD_URL
    fibo-fnd-utl-alx:hasArgument ?cb_ipo_uuid_pricePerShare_USD_URL;
    fibo-fnd-dt-fd:hasObservedDateTime ?went_public_on_xsd_date;
    fibo-fnd-rel-rel:appliesTo ?cb_agent_org_uuid_issuer_URL.
  ?cb_ipo_uuid_marketCapValue_valuation_price_currency_code_URL a fibo-fnd-acc-cur:Mo
    rdfs:label "Monetary amount of Market Cap (valuation) in local currency";
    fibo-fnd-acc-cur:hasAmount ?valuation_price_xsd_decimal;
    fibo-fnd-acc-cur:hasCurrency ?cb_currency_valuation_price_currency_code_URL.
  ?cb_ipo_uuid_marketCapValue_USD_URL a fibo-fnd-acc-cur:MonetaryAmount;
    rdfs:label "Monetary amount of Market Cap (valuation) in USD";
    fibo-fnd-acc-cur:hasAmount ?valuation_price_usd_xsd_decimal;
    fibo-fnd-acc-cur:hasCurrency <cb/currency/USD>.
  ?cb_ipo_uuid_pricePerShare_share_price_currency_code_URL a fibo-sec-eq-eq:PricePerS
    rdfs:label "Price per share at the time of IPO (permalink) in local currency";
    fibo-fnd-acc-cur:hasAmount ?share_price_xsd_decimal;
    fibo-fnd-acc-cur:hasCurrency ?cb_currency_share_price_currency_code_URL;
    fibo-fnd-dt-fd:hasObservedDateTime ?went_public_on_xsd_date;
    fibo-fnd-acc-cur:isPriceFor ?cb_ipo_uuid_share_URL.
  ?cb_ipo_uuid_pricePerShare_USD_URL a fibo-sec-eq-eq:PricePerShare;
    rdfs:label "Price per share at the time of IPO (permalink) in USD";
    fibo-fnd-acc-cur:hasAmount ?share_price_usd_xsd_decimal;
    fibo-fnd-acc-cur:hasCurrency <cb/currency/USD>;
    fibo-fnd-dt-fd:hasObservedDateTime ?went_public_on_xsd_date;
    fibo-fnd-acc-cur:isPriceFor ?cb_ipo_uuid_share_URL.
  ## Currencies: USD plus 3 more for the 3 financials
  <cb/currency/USD> a fibo-fnd-acc-cur:Currency;
    lcc-lr:hasName 'USD';
    rdfs:label 'Currency USD'.
  <cb/currency/USD/code> a fibo-fnd-acc-cur:CurrencyIdentifier;
    rdfs:label 'Code of currency USD';
    lcc-lr:hasTag 'USD';
    lcc-lr:denotes <cb/currency/USD>;
    lcc-lr:identifies <cb/currency/USD>;
    lcc-lr:isMemberOf <cb/currency>.
  ?cb_currency_share_price_currency_code_URL a fibo-fnd-acc-cur:Currency;
    lcc-lr:hasName ?share_price_currency_code;
    rdfs:label 'Currency (share_price_currency_code)'.
  ?cb_currency_share_price_currency_code_code_URL a fibo-fnd-acc-cur:CurrencyIdentifi
    rdfs:label 'Code of currency (share_price_currency_code)';
    lcc-lr:hasTag ?share_price_currency_code;
    lcc-lr:denotes ?cb_currency_share_price_currency_code_URL;
    lcc-lr:identifies ?cb_currency_share_price_currency_code_URL;
    lcc-lr:isMemberOf <cb/currency>.
  ?cb_currency_valuation_price_currency_code_URL a fibo-fnd-acc-cur:Currency;
    lcc-lr:hasName ?valuation_price_currency_code;
    rdfs:label 'Currency (valuation_price_currency_code)'.
  ?cb_currency_valuation_price_currency_code_code_URL a fibo-fnd-acc-cur:CurrencyIden
    rdfs:label 'Code of currency (valuation_price_currency_code)';
    lcc-lr:hasTag ?valuation_price_currency_code;
    lcc-lr:denotes ?cb_currency_valuation_price_currency_code_URL;
    lcc-lr:identifies ?cb_currency_valuation_price_currency_code_URL;
    lcc-lr:isMemberOf <cb/currency>.
  ?cb_currency_money_raised_currency_code_URL a fibo-fnd-acc-cur:Currency;
    lcc-lr:hasName ?money_raised_currency_code;
    rdfs:label 'Currency (money_raised_currency_code)'.
  ?cb_currency_money_raised_currency_code_code_URL a fibo-fnd-acc-cur:CurrencyIdentif
    rdfs:label 'Code of currency (money_raised_currency_code)';
    lcc-lr:hasTag ?money_raised_currency_code;
    lcc-lr:denotes ?cb_currency_money_raised_currency_code_URL;
    lcc-lr:identifies ?cb_currency_money_raised_currency_code_URL;
    lcc-lr:isMemberOf <cb/currency>.
  <cb/currency> a lcc-lr:IdentificationScheme, lcc-lr:CodeSet;
    rdfs:label 'CrunchBase currency code set';
    skos:definition 'Currency identifiers used in CrunchBase';
    skos:scopeNote 'Better reconcile to fibo-fnd-acc-4217:ISO4217-CodeSet'.
}}
where {
  service <rdf-mapper:ontorefine:PROJECT_ID> {
    bind(?c_org_uuid as ?org_uuid)
    bind(?c_uuid as ?uuid)
    bind(?c_stock_exchange_symbol as ?stock_exchange_symbol)
    bind(?c_permalink as ?permalink)
    bind(?c_money_raised_currency_code as ?money_raised_currency_code)
    bind(?c_went_public_on as ?went_public_on)
    bind(?c_name as ?name)
    bind(?c_cb_url as ?cb_url)
    bind(?c_rank as ?rank)
    bind(?c_created_at as ?created_at)
    bind(?c_updated_at as ?updated_at)
    bind(?c_stock_symbol as ?stock_symbol)
    bind(?c_share_price_currency_code as ?share_price_currency_code)
    bind(?c_money_raised as ?money_raised)
    bind(?c_money_raised_usd as ?money_raised_usd)
    bind(?c_valuation_price_currency_code as ?valuation_price_currency_code)
    bind(?c_valuation as ?valuation)
    bind(?c_valuation_price as ?valuation_price)
    bind(?c_valuation_price_usd as ?valuation_price_usd)
    bind(?c_share_price as ?share_price)
    bind(?c_share_price_usd as ?share_price_usd)
    bind(iri(concat("cb/agent/",?org_uuid,"/issuer")) as ?cb_agent_org_uuid_issuer_UR
    bind(iri(concat("cb/ipo/",?uuid,"/share")) as ?cb_ipo_uuid_share_URL)
    bind(iri(concat("cb/agent/",?org_uuid)) as ?cb_agent_org_uuid_URL)
    bind(iri(concat("cb/exchange/",?stock_exchange_symbol)) as ?cb_exchange_stock_exc
    bind(iri(concat("cb/exchange/",?stock_exchange_symbol,"/code")) as ?cb_exchange_s
    bind(iri(concat("cb/ipo/",?uuid,"/offering")) as ?cb_ipo_uuid_offering_URL)
    bind(iri(concat("cb/ipo/",?uuid,"/price/",?money_raised_currency_code)) as ?cb_ip
    bind(iri(concat("cb/ipo/",?uuid,"/price/USD")) as ?cb_ipo_uuid_price_USD_URL)
    bind(strdt(?went_public_on,xsd:date) as ?went_public_on_xsd_date)
    bind(strdt(?cb_url,xsd:anyURI) as ?cb_url_xsd_anyURI)
    bind(strdt(?rank,xsd:integer) as ?rank_xsd_integer)
    bind(REPLACE(?created_at,' ','T') as ?created_at_FIXDATE)
    bind(strdt(?created_at_FIXDATE,xsd:dateTime) as ?created_at_FIXDATE_xsd_dateTime)
    bind(REPLACE(?updated_at,' ','T') as ?updated_at_FIXDATE)
    bind(strdt(?updated_at_FIXDATE,xsd:dateTime) as ?updated_at_FIXDATE_xsd_dateTime)
    bind(iri(concat("cb/ipo/",?uuid,"/listing")) as ?cb_ipo_uuid_listing_URL)
    bind(iri(concat("cb/currency/",?share_price_currency_code)) as ?cb_currency_share
    bind(iri(concat("cb/ipo/",?uuid,"/ticker")) as ?cb_ipo_uuid_ticker_URL)
    bind(strdt(?money_raised,xsd:decimal) as ?money_raised_xsd_decimal)
    bind(iri(concat("cb/currency/",?money_raised_currency_code)) as ?cb_currency_mone
    bind(strdt(?money_raised_usd,xsd:decimal) as ?money_raised_usd_xsd_decimal)
    bind(iri(concat("cb/ipo/",?uuid,"/marketCap/",?valuation_price_currency_code)) as
    bind(iri(concat("cb/ipo/",?uuid,"/marketCapValue/",?valuation_price_currency_code
    bind(iri(concat("cb/ipo/",?uuid,"/pricePerShare/",?share_price_currency_code)) as
    bind(iri(concat("cb/ipo/",?uuid,"/marketCap/USD")) as ?cb_ipo_uuid_marketCap_USD_
    bind(iri(concat("cb/ipo/",?uuid,"/marketCapValue/USD")) as ?cb_ipo_uuid_marketCap
    bind(iri(concat("cb/ipo/",?uuid,"/pricePerShare/USD")) as ?cb_ipo_uuid_pricePerSh
    bind(strdt(?valuation_price,xsd:decimal) as ?valuation_price_xsd_decimal)
    bind(iri(concat("cb/currency/",?valuation_price_currency_code)) as ?cb_currency_v
    bind(strdt(?valuation_price_usd,xsd:decimal) as ?valuation_price_usd_xsd_decimal)
```

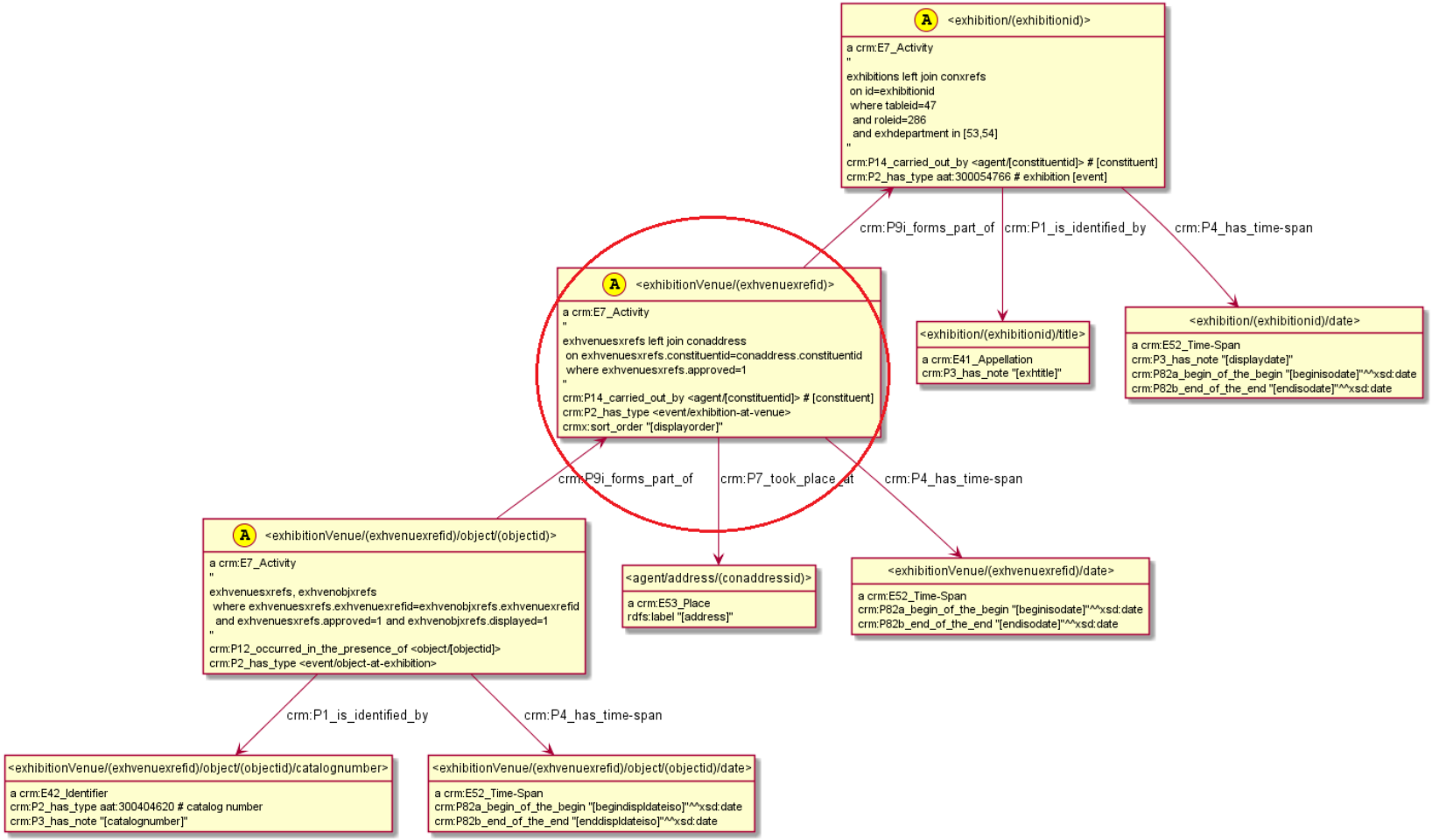
Generating R2RML Transforms

R2RML Transforms From Turtle Examples

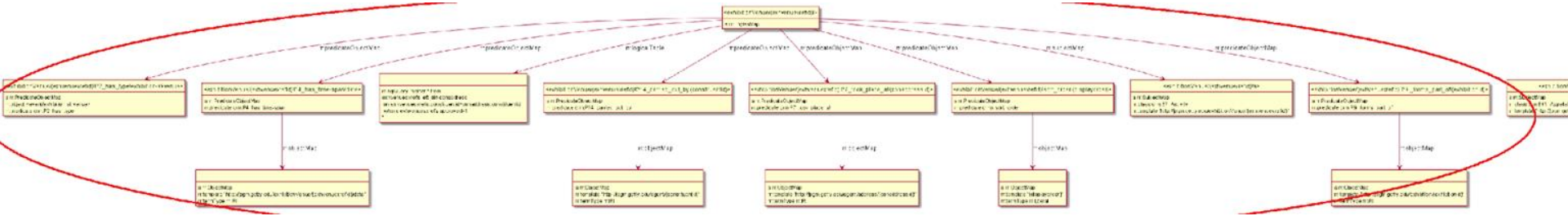
- **R2RML**: W3C language to map RDBMS to RDF
- **rdf2rml**: generates R2RML transforms from Turtle examples
  - Table/join in node comment; carried to “children” nodes
  - Field in attribute/URL
  - Both the model and R2RML are RDF, so implemented as some SPARQL transformation
  - TODO: implement functions
  - Example: Exhibitions from J.P.Getty Museum
- Then can use any R2RML implementation:
  - ONTOP for RDBMS Virtualization (OBDA)
  - GraphDB Virtual Repositories for Virtualization or Materialization
  - Various implementations, see awesome-semantic-web: r2rml:
    Antidot d2btriples, Morph-KGC, RMLmapper, RocketRML, carml…



Getty Exhibitions: Turtle model



Getty Exhibitions: Generated R2RML



- R2RML is quite verbose:
  - subject map,
  - predicateObject map,
  - Templates and details about each URL and literal, etc.
- The one circled node is expanded to 15 nodes

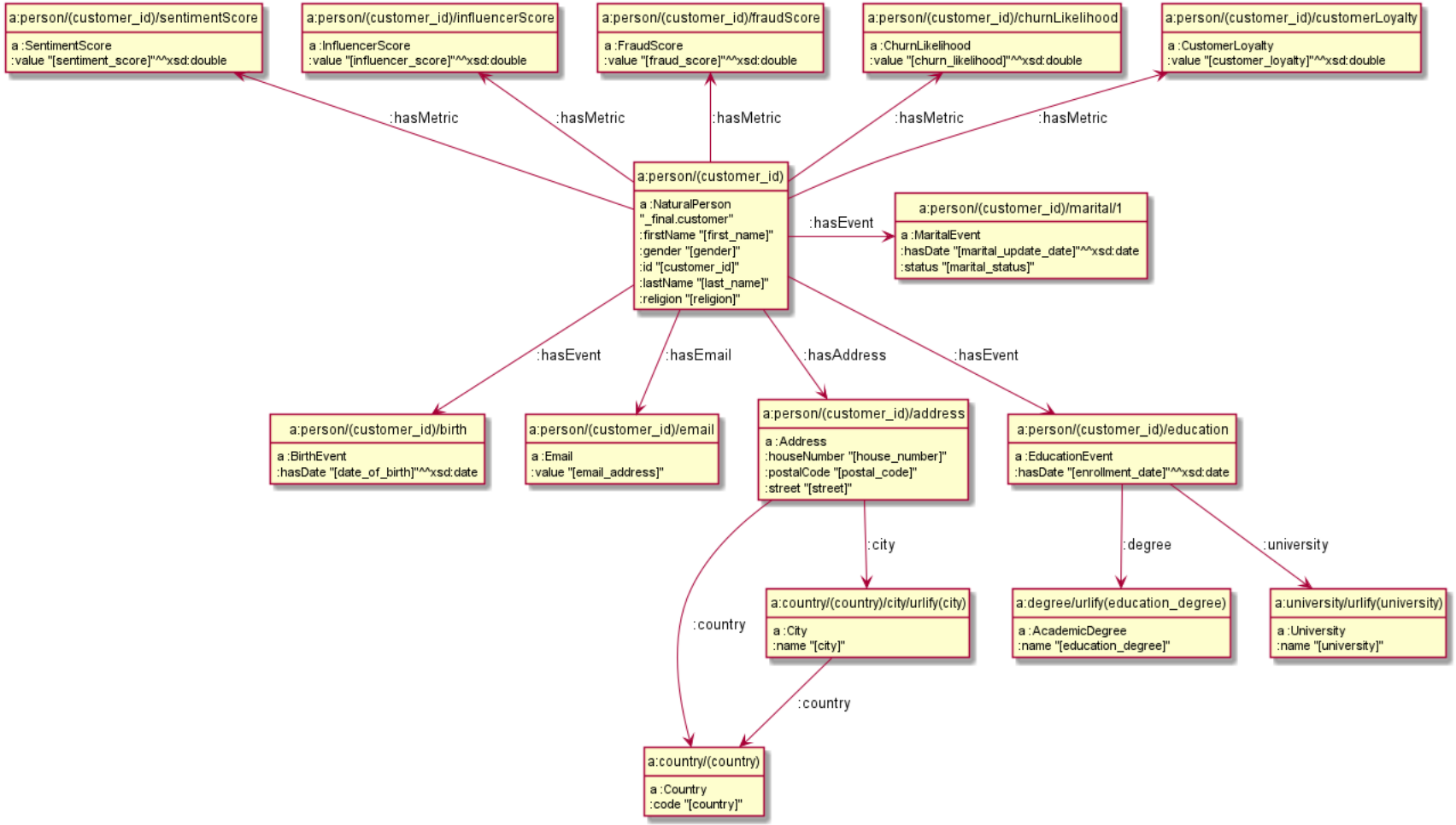
Getty Exhibitions: Model (60 lines) vs R2RML (315 lines)

```
# LEVEL 2: EXHIBITION AT VENUE
# An exhibition may visit a venue more than once: MUSLOD-1
<exhibitionVenue/(exhvenuexrefid)>
  puml:label """
exhvenuesxrefs left join conaddress
 on exhvenuesxrefs.constituentid=conaddress.constituentid
 where exhvenuesxrefs.approved=1
""";
  a crm:E7_Activity;
  crm:P2_has_type <event/exhibition-at-venue>;
  crm:P9i_forms_part_of <exhibition/(exhibitionid)>;
  crm:P14_carried_out_by <agent/(constituentid)>;
  crm:P7_took_place_at <agent/address/(conaddressid)>;
  crm:P4_has_time-span <exhibitionVenue/(exhvenuexrefid)/date>;
  crmx:sort_order "(displayorder)".

<exhibitionVenue/(exhvenuexrefid)/date> a crm:E52_Time-Span;
  crm:P82a_begin_of_the_begin "(beginisodate)"^^xsd:date;
  crm:P82b_end_of_the_end "(endisodate)"^^xsd:date.

<agent/(constituentid)> a crm:E39_Actor;
  rdfs:label "(constituent)".

<agent/address/(conaddressid)> a crm:E53_Place;
  rdfs:label "(address)".

## LEVEL 3: OBJECT AT EXHIBITION AT VENUE
<exhibitionVenue/(exhvenuexrefid)/object/(objectid)>
  puml:label """
exhvenuesxrefs, exhvenobjxrefs
 where exhvenuesxrefs.exhvenuexrefid=exhvenobjxrefs.exhvenuexrefid
  and exhvenuesxrefs.approved=1 and exhvenobjxrefs.displayed=1
""";
  a crm:E7_Activity;
  crm:P2_has_type <event/object-at-exhibition>;
  crm:P9i_forms_part_of <exhibitionVenue/(exhvenuexrefid)>;
  crm:P12_occurred_in_the_presence_of <object/(objectid)>;
  crm:P4_has_time-span <exhibitionVenue/(exhvenuexrefid)/object/(objectid)/date>;
  crm:P1_is_identified_by <exhibitionVenue/(exhvenuexrefid)/object/(objectid)/catalognumber>.

<exhibitionVenue/(exhvenuexrefid)/object/(objectid)/date> a crm:E52_Time-Span;
  crm:P82a_begin_of_the_begin "(begindisplдateiso)"^^xsd:date;
  crm:P82b_end_of_the_end "(enddisplдateiso)"^^xsd:date.

<exhibitionVenue/(exhvenuexrefid)/object/(objectid)/catalognumber> a crm:E42_Identifier;
  crm:P2_has_type aat:300404620; # catalog number: ITSLOD-466
  crm:P3_has_note "(catalognumber)".

####################
aat:300054766 a puml:Inline; skos:prefLabel "exhibition (event)".
aat:300404620 a puml:Inline; skos:prefLabel "catalog number".
crm:P9i_forms_part_of puml:arrow puml:up.
<agent/(constituentid)> a puml:Inline.
<event/exhibition-at-venue> a puml:Inline.
<event/object-at-exhibition> a puml:Inline.
```

```
<agent/address/(conaddressid)!label!(address)>
        a               rr:PredicateObjectMap ;
        rr:objectMap    [ a             rr:ObjectMap ;
                          rr:template   "{address}" ;
                          rr:termType   rr:Literal
                        ] ;
        rr:predicate    rdfs:label .

<agent/address/(conaddressid)!map>
        a                       rr:TriplesMap ;
        rr:logicalTable         [ rr:sqlQuery  "select * from
\nexhvenuesxrefs left join conaddress \n on
exhvenuesxrefs.constituentid=conaddress.constituentid  \n where
exhvenuesxrefs.approved=1\n" ] ;
        rr:predicateObjectMap
<agent/address/(conaddressid)!label!(address)> ;
        rr:subjectMap           <agent/address/(conaddressid)!subj> .

<agent/address/(conaddressid)!subj>
        a               rr:SubjectMap ;
        rr:class        crm:E53_Place ;
        rr:template     "agent/address/{conaddressid}" .

<exhibition/(exhibitionid)!P14_carried_out_by!(constituentid)>
        a               rr:PredicateObjectMap ;
        rr:objectMap    [ a             rr:ObjectMap ;
                          rr:template   "agent/{constituentid}" ;
                          rr:termType   rr:IRI
                        ] ;
        rr:predicate    crm:P14_carried_out_by .

<exhibition/(exhibitionid)!P1_is_identified_by!title>
        a               rr:PredicateObjectMap ;
        rr:objectMap    [ a             rr:ObjectMap ;
                          rr:template   "exhibition/{exhibitionid}/title" ;
                          rr:termType   rr:IRI
                        ] ;
        rr:predicate    crm:P1_is_identified_by .

<exhibition/(exhibitionid)!P2_has_type!300054766>
        a               rr:PredicateObjectMap ;
        rr:object       aat:300054766 ;
        rr:predicate    crm:P2_has_type .
<exhibition/(exhibitionid)!P4_has_time-span!date>
        a               rr:PredicateObjectMap ;
        rr:objectMap    [ a             rr:ObjectMap ;
                          rr:template   "exhibition/{exhibitionid}/date" ;
                          rr:termType   rr:IRI
                        ] ;
        rr:predicate    crm:P4_has_time-span .

<exhibition/(exhibitionid)!map>
        a                       rr:TriplesMap ;
        rr:logicalTable         [ rr:sqlQuery  "select * from
\nexhibitions left join conxrefs \n on id=exhibitionid\n where
```

```
        rr:logicalTable         [ rr:sqlQuery  "select * from
\nexhibitions left join conxrefs \n on id=exhibitionid\n where
tableid=47\n  and roleid=286  \n  and exhdepartment in (53,54)\n" ] ;
        rr:predicateObjectMap
<exhibition/(exhibitionid)!P1_is_identified_by!title> ,
<exhibition/(exhibitionid)!P4_has_time-span!date> ,
<exhibition/(exhibitionid)!P14_carried_out_by!(constituentid)> ,
<exhibition/(exhibitionid)!P2_has_type!300054766> ;
        rr:subjectMap           <exhibition/(exhibitionid)!subj> .

<exhibition/(exhibitionid)!subj>
        a               rr:SubjectMap ;
        rr:class        crm:E7_Activity ;
        rr:template     "exhibition/{exhibitionid}" .

<exhibition/(exhibitionid)/date!P3_has_note!(displaydate)>
        a               rr:PredicateObjectMap ;
        rr:objectMap    [ a             rr:ObjectMap ;
                          rr:template   "{displaydate}" ;
                          rr:termType   rr:Literal
                        ] ;
        rr:predicate    crm:P3_has_note .

<exhibition/(exhibitionid)/date!P82a_begin_of_the_begin!(beginisodate)>
        a               rr:PredicateObjectMap ;
        rr:objectMap    [ a             rr:ObjectMap ;
                          rr:datatype   xsd:date ;
                          rr:template   "{beginisodate}" ;
                          rr:termType   rr:Literal
                        ] ;
        rr:predicate    crm:P82a_begin_of_the_begin .

<exhibition/(exhibitionid)/date!P82b_end_of_the_end!(endisodate)>
        a               rr:PredicateObjectMap ;
        rr:objectMap    [ a             rr:ObjectMap ;
                          rr:datatype   xsd:date ;
                          rr:template   "{endisodate}" ;
                          rr:termType   rr:Literal
                        ] ;
        rr:predicate    crm:P82b_end_of_the_end .

<exhibition/(exhibitionid)/date!map>
        a                       rr:TriplesMap ;
        rr:logicalTable         [ rr:sqlQuery  "select * from
\nexhibitions left join conxrefs \n on id=exhibitionid\n where
tableid=47\n  and roleid=286  \n  and exhdepartment in (53,54)\n" ] ;
        rr:predicateObjectMap
<exhibition/(exhibitionid)/date!P82b_end_of_the_end!(endisodate)> ,
<exhibition/(exhibitionid)/date!P3_has_note!(displaydate)> ,
<exhibition/(exhibitionid)/date!P82a_begin_of_the_begin!(beginisodate)> ;
        rr:subjectMap           <exhibition/(exhibitionid)/date!subj> .

<exhibition/(exhibitionid)/date!subj>
        a               rr:SubjectMap ;
        rr:class        crm:E52_Time-Span ;
```

Insurance Customer Example: Model



Insurance Example: Model (52 lines) vs R2RML (502 lines)

Backup slides

Will present them if there is time

Publications Office of the European Union

interoperable europe

innovation ∞ govtech ∞ community

Semantic ETL Desiderata

Main requirements for semantic ETL
(green: yes or positive, orange: maybe or warning, red: not or negative)

ETL and Modeling

- **Declarative approach**: conversion scripts or queries should be written in a high-level language (such as SPARQL, XSPARQL, YARRML, LIXR). This ensures correctness and easier maintainability.
- **Generation**: conversion scripts should be generated from semantic models, which are also used to generate validation scripts. This saves further effort, ensures correctness and easier maintainability.
- **Polyglot Modeling**: use the same set of models to generate a variety of technical artefacts, ensuring they are consistent and up to date: conversions, validations, JSON-LD contexts and frames, UI forms, etc.

ETL Performance

- Streaming/chunking:
  - Tools that collect RDF output to a memory RDF model and then output it only work for thousands of records.
  - To work for millions records, the tool must output each piece of RDF as soon as it's ready, otherwise memory requirements and execution time grow dramatically. OntoRefine and TARQL do.
- One pass: ETL tools should read the input data only once.
  - OntoRefine does (on simple tabular data).
  - R2RML mappers should process all maps applying to the same query/table at once for each row, instead of iterating over the same resultset/table multiple times.
- Direct update:
  - ETL tool should be able to output directly to a semantic repository (e.g. GraphDB), both for ingest and update. OntoRefine does, TARQL doesn't.
  - This also allows to join to existing semantic data, e.g. to resolve keywords to nomenclature nodes.

ETL Logistics

- **SRGO Methodology**
  - Steps, Reasoning, Repos/Graphs, Ordering, Ontologies, Reasoning.
  - Strictly describe the What, Where, When, How of a pipeline.
- Prerequisite for:
  - Making the pipeline reproducible
  - Making the KG building process sustainable in the long run
- **What**: are the steps of the pipeline process.
  - **RDF** (usually Turtle): obtained from LOD, written by hand (small), converted once (static)
  - **Non-RDF** data: where to get them from, what credentials are needed
  - **Ingestion scripts** (declarative or procedural)
  - **Virtualization transforms**: ONTOP or R2RML mappings; MongoDB JSON connector
  - **Transformation/correction**: SPARQL Updates that can transform data, derive new data, add missing data, correct fields
  - **Validation steps**: SHACL, SHEX or SPARQL validations

ETL Logistics (2)

- **Where** the steps take place:
  - Several **Repositories** for data separation or independent update cycles
    - GraphDB Internal Federation eliminates network overhead and improves security
  - **Named Graphs** for:
    - Unit of Work
    - Idempotence
    - Unit of Transfer
    - Provenance
- **When** each step is executed:
  - The graph nature of RDF and GraphDB's incremental inference ensure that RDF steps are compositional, i.e. can be done in any order
  - Other steps have ordering requirements, which are usually sparse. This partial order needs to be resolved to a total order for step execution
- **How**: what inference is needed by the pipeline
  - **Reasoning ruleset**: RDFS, RDFS+, OWL-Horst, OWL RL, OWL QL, etc
  - **Ontologies loaded**. Many ontologies come with heavy inference (part of heavy "ontological commitment") that you may not need

ETL Predictability and Transactionality

- **Locality**:
  - Each ETL step should operate in its own graph and should not disturb any data that is the responsibility of other steps.
  - Depends on ETL logistics and using named graphs
- **Predictability and Idempotence**:
  - The results of each step should not depend on the previous state of its own graph (but can depend on the data of previous steps), in order to be predictable and reproducible.
  - Running the same step twice should produce the same result (idempotence). Both satisfied by OntoRefine with UPDATE queries.
- **Transactionality**:
  - Each step should leave the total KG operational and with full data, no matter how long it takes.
  - ETL should use transactional (ACID) semantics or use a staging data store, in order to avoid prolonged down-time of the KG.

ETL Virtualization and Automation

- **NoETL**: if source data is very large (Volume) or changes very frequently (Velocity), it may be better to use data in its original form and provide SPARQL querying over it, rather than converting it to RDF
  - Relational Virtualization using ONTOP
  - JSON "virtualization" using MongoDB Integration
- **Orchestration**: for complex ETL pipelines or when one needs to track multiple ETL instances
  - Use orchestration tools like CWL, Airflow, Apache HOP, etc
- **Automation**: to ensure reproducibility in different developer and production environments, package up complex steps and whole pipelines
  - Using automated deployment tools like Docker, Kubernetes, Helm.