Key Things You Need to Know About RDF

and Why They Are Important

David Booth, Ph.D.
Hawaii Resource Group
david@dbooth.org

Semantic Technology and Business Conference
21-Aug-2014

Latest version of these slides:
http://dbooth.org/2014/key/

RDF is

fundamentally different

from other data formats – XML, JSON, etc.

This presentation explains why.

But first, some background . . .

Comparing RDF with XML or JSON

WARNING: Improper comparison!

- XML, JSON or any other format could be used in special ways to achieve all of RDF's features
  - But that isn't how they are normally used
- This talk compares RDF with XML and JSON *as they are normally used*

What is RDF?

- "Resource Description Framework"
  - *But think "Reusable Data Framework"*
- Language for representing information
- Vendor-neutral international standard by W3C
- Mature – 10+ years
- Used in many domains, including biomedical and pharma

RDF graph

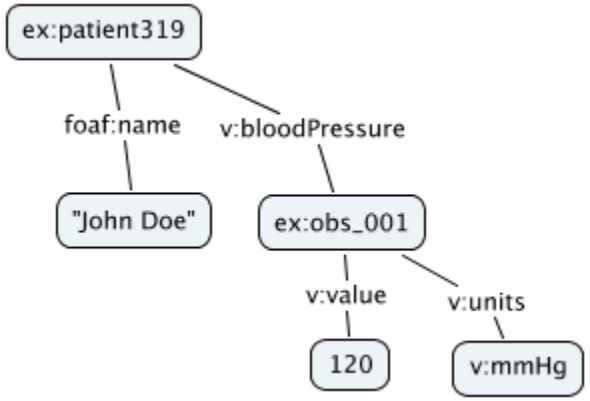
English assertions:

Patient319 has name "John Doe".
Patient319 has systolic blood pressure observation Obs_001.
Obs_001 value was 120.
Obs_001 units was mmHg.

RDF* assertions ("triples"):

ex:patient319 foaf:name "John Doe" .
ex:patient319 v:systolicBP ex:obs_001 .
ex:obs_001 v:value 120 .
ex:obs_001 v:units v:mmHg .

RDF graph:



*Namespace definitions omitted

What is RDF good for?

- Large-scale information integration
- Semantically connecting diverse data models and vocabularies
- Translating between data models and vocabularies
- Smarter data use

Let's see why . . .

Key things you need to know about RDF

#5: RDF is self describing

- RDF uses URIs as identifiers

#4: RDF is easy to map from other data representations

- RDF data is made of assertions

#3: RDF captures information – not syntax

- RDF is format independent

#2: Multiple data models and vocabularies can be easily combined and interrelated

- RDF is multi-schema friendly

#1: RDF enables smarter data use and automated data translation

- RDF enables inference

#5: RDF is self describing

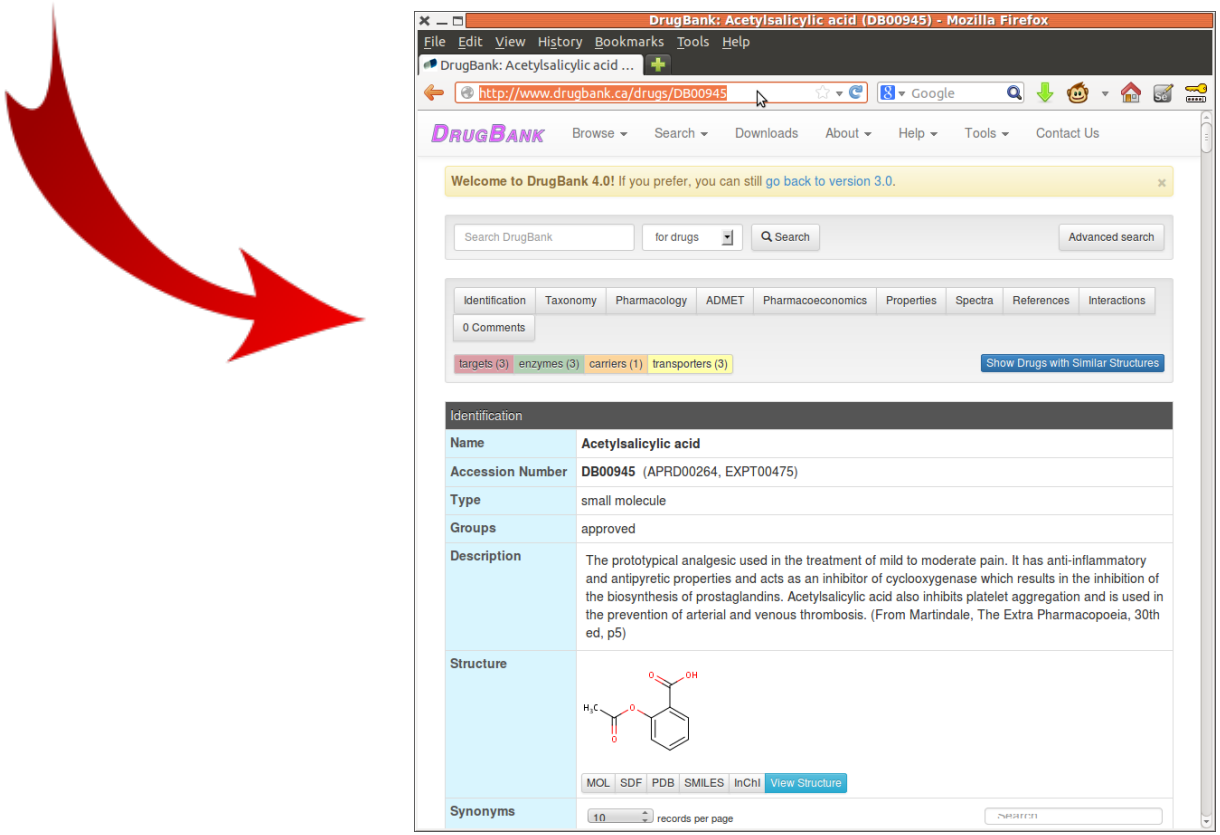
- RDF uses URIs as identifiers
- Terms, data models, properties, vocabularies, etc. – almost everything
  - E.g., identifier for aspirin: <http://www.drugbank.ca/drugs/DB00945>
- URIs can be abbreviated:

  @prefix db: <http://www.drugbank.ca/drugs/> .

  . . . db:DB00945 . . .

Example: URI for Aspirin

http://www.drugbank.ca/drugs/DB00945



Why is this important?

- Enables unambiguous identifiers without the bottleneck of central control
  - New URIs can be created by any party
- Web friendly: URI can link to an authoritative definition
- Linking to definition is a best practice – not an RDF requirement
  - A/k/a "Linked Data"

What if the URI cannot be dereferenced?

- Then the definition must be found some other way
  - (Just as with current medical codes)

Why is this important?

- Terms in a vocabulary can be **self-describing**
  - Authoritative definition can be easily located
  - Reduces ambiguity
- For standard terms this is a convenience
- For non-standard terms:
  - Enables definition to be found by any party
  - Aids in bootstrapping new terms toward standardization

Supports standards and diversity

Terms are **self describing**?

- XML:
  - Can be just as good as RDF if namespaces are properly used
  - In practice, namespaces are not always used or clickable to definitions
- JSON:
  - In theory, could be used like RDF
  - In practice, almost never done

#4: RDF is easy to map from other data representations

- RDF is made up of lots of small, atomic statements, called *assertions* or *triples*
- Each assertion is a triple, like *subject-verb-object* of a simple sentence
- Set of assertions is called an *RDF graph*
  - Nodes are subjects and objects

Single RDF assertion / triple

English:

Patient319 has name "John Doe".

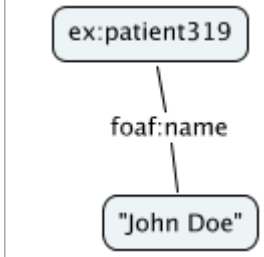
Subject *Verb phrase* *Object*

RDF:

ex:patient319 foaf:name "John Doe" .

Subject *Predicate** *Object***

RDF graph:



*A/k/a property or relation
**A/k/a value

RDF assertions form graphs

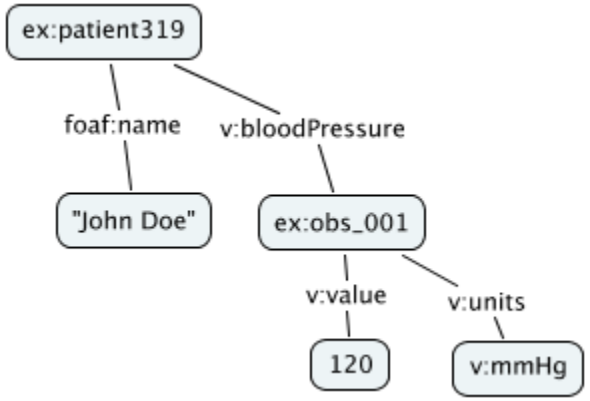
English assertions:

Patient319 has name "John Doe".
Patient319 has systolic blood pressure observation Obs_001.
Obs_001 value was 120.
Obs_001 units was mmHg.

RDF assertions ("triples"):

ex:patient319 foaf:name "John Doe" .
ex:patient319 v:systolicBP ex:obs_001 .
ex:obs_001 v:value 120 .
ex:obs_001 v:units v:mmHg .

RDF graph:

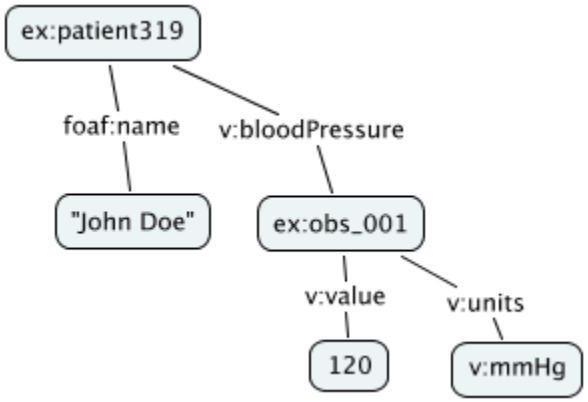


Why does this matter?

- Easy to represent any data
- Easy to incorporate any data model
  - Hierarchical, relational, graph, etc.

Great for data integration!

Hierarchical data model in RDF



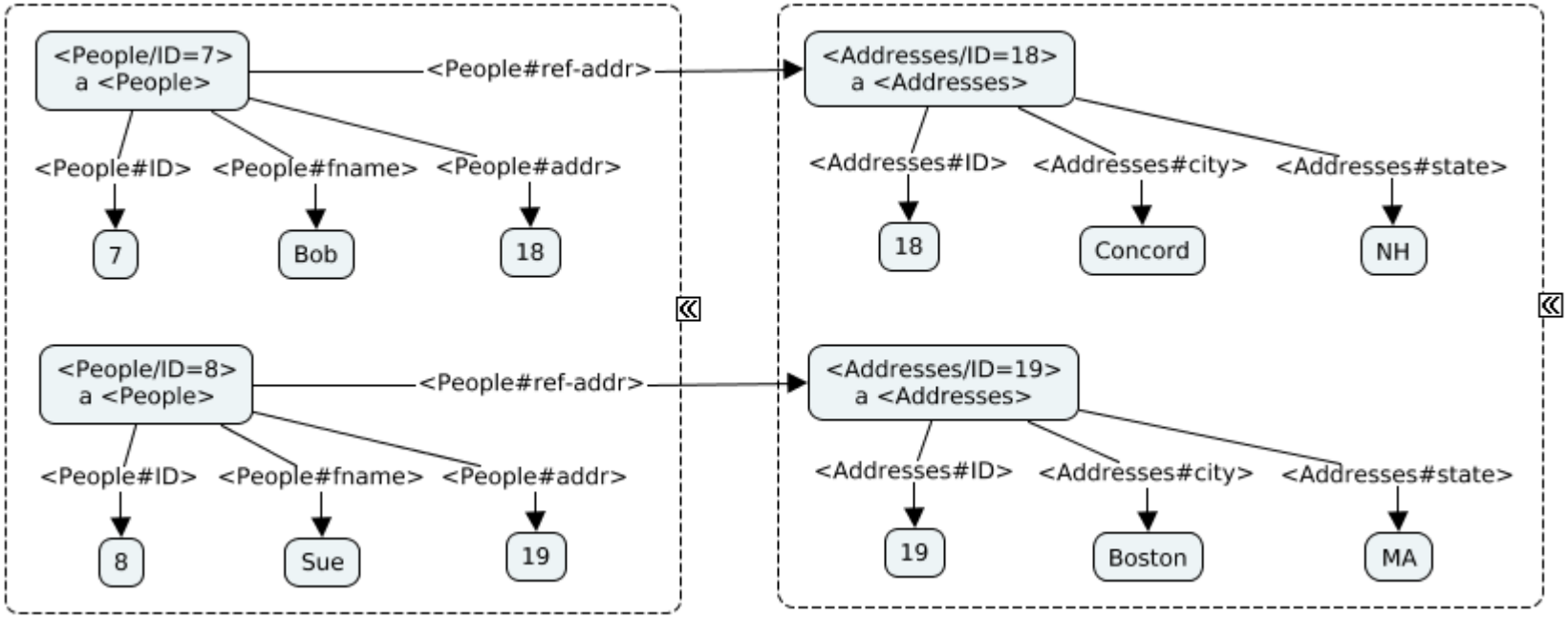
Relational data model in RDF

People

ID	fname	addr
7	Bob	18
8	Sue	19

Addresses

ID	City	State
18	Concord	NH
19	Boston	MA



See W3C Direct Mapping of Relational Data to RDF:
http://www.w3.org/TR/rdb-direct-mapping/

Why does this matter?

- Easy to map any data format to RDF
  - E.g., XML, JSON, CSV, SQL tables, etc.

Easy to map from other formats?

- XML:
  - Except cyclic graphs

- JSON:
  - Except cyclic graphs

#3: RDF captures information – not syntax

- RDF is format independent
- There are multiple RDF syntaxes: Turtle, N-Triples, JSON-LD, RDF/XML, etc.
- The same information can be written in different formats

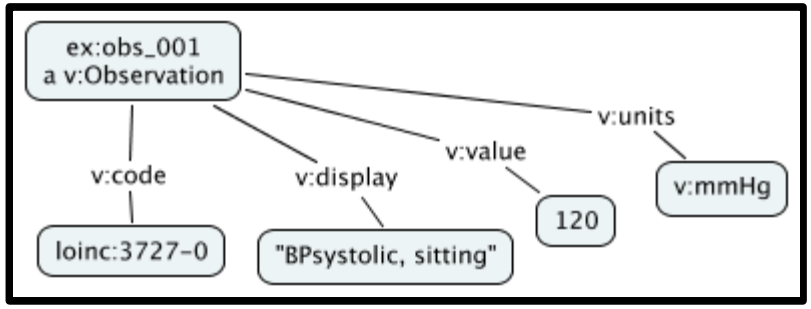
RDF examples

RDF (Turtle)

```
@prefix ex: <http://example/ex/> .
@prefix loinc: <http://loinc.org/> .
@prefix v: <http://example/v/> .

ex:obs_001 a v:Observation ;
 v:code loinc:3727-0 ;
 v:display "BPsystolic, sitting" ;
 v:value 120 ;
 v:units v:mmHg .
```

RDF graph



RDF (N-Triples)

```
<http://example/ex/obs_001> <http://www.w3.org/1999/02/22-rdf-syntax-ns#type> <http://example/v/Observation> .
<http://example/ex/obs_001> <http://example/v/code> <http://loinc.org/3727-0> .
<http://example/ex/obs_001> <http://example/v/display> "BPsystolic, sitting" .
<http://example/ex/obs_001> <http://example/v/value> "120"^^<http://www.w3.org/2001/XMLSchema#integer> .
<http://example/ex/obs_001> <http://example/v/units> <http://example/v/mmHg> .
```

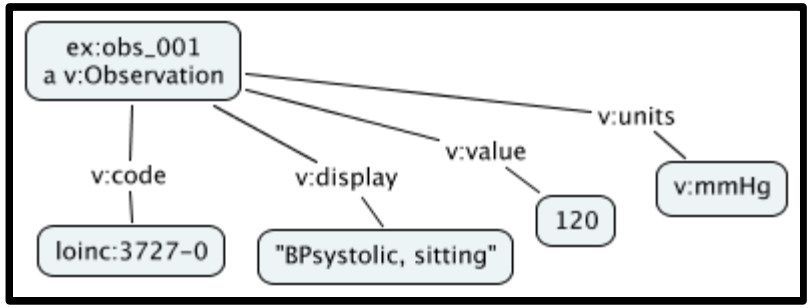
Same information!

RDF examples

RDF (JSON-LD)

```
{
  "@id": "http://example/ex/obs_001",
  "@type": "http://example/v/Observation",
  "http://example/v/code": {
    "@id": "http://loinc.org/3727-0"
  },
  "http://example/v/display": "BPsystolic, sitting",
  "http://example/v/units": {
    "@id": "http://example/v/mmHg"
  },
  "http://example/v/value": 120
}
```

RDF graph



RDF (RDF/XML)

```
<?xml version="1.0" encoding="utf-8"?>
<rdf:RDF xmlns:ex="http://example/ex/" xmlns:loinc="http://loinc.org/"
      xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#" xmlns:v="http://example/v/">
  <rdf:Description rdf:about="http://example/ex/obs_001">
    <rdf:type rdf:resource="http://example/v/Observation"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://example/ex/obs_001">
    <v:code rdf:resource="http://loinc.org/3727-0"/>
  </rdf:Description>
  <rdf:Description rdf:about="http://example/ex/obs_001">
    <v:display>BPsystolic, sitting</v:display>
  </rdf:Description>
  <rdf:Description rdf:about="http://example/ex/obs_001">
    <v:value rdf:datatype="http://www.w3.org/2001/XMLSchema#integer">120</v:value>
  </rdf:Description>
  <rdf:Description rdf:about="http://example/ex/obs_001">
    <v:units rdf:resource="http://example/v/mmHg"/>
  </rdf:Description>
</rdf:RDF>
```

Same info!

Why does this matter?

- Emphasis is on the meaning (where it should be)
- RDF can be used to capture the meaning of other data formats/languages:
  - Any data format can be mapped to RDF to capture its meaning
  - RDF acts as a substrate language

Different source languages, same RDF

HL7 v2.x

```
OBX|1|CE|3727-0^BPsystolic,
sitting||120||mmHg|
```

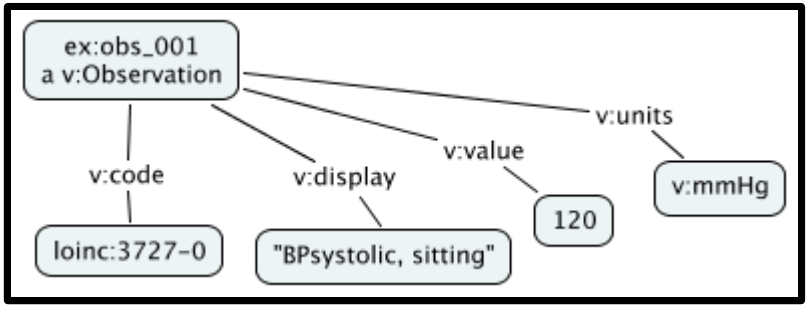
Maps to

FHIR

```
<Observation
      xmlns="http://hl7.org/fhir">
   <system value="http://loinc.org"/>
   <code value="3727-0"/>
   <display value="BPsystolic, sitting"/>
   <value value="120"/>
   <units value="mmHg"/>
</Observation>
```

Maps to

RDF graph



Why does this matter?

- Precise meaning of data in other languages/formats can be captured in a consistent, format-independent way
- Important for data integration

Captures meaning, not syntax?

- XML:
  - Syntax only

- JSON:
  - Syntax only

½

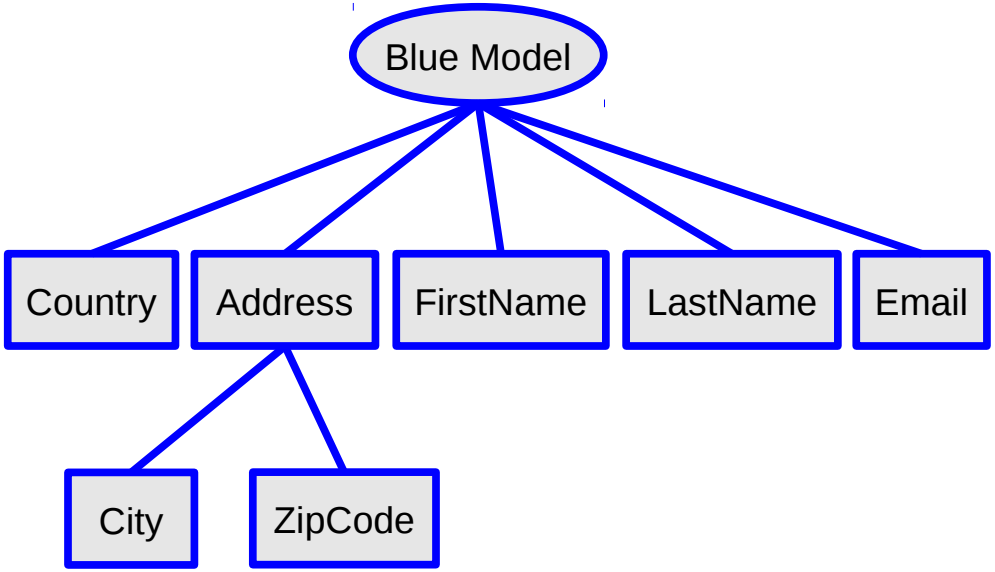
#2: Multiple data models and vocabularies can be easily combined and interrelated

- RDF is multi-schema friendly*
  - (In this talk, schema == data model, i.e., the shape of the data)
- Multiple data models/schemas and vocabularies can peacefully co-exist, semantically connected

*A/k/a schema-promiscuous, schema-flexible, schema-less, etc.

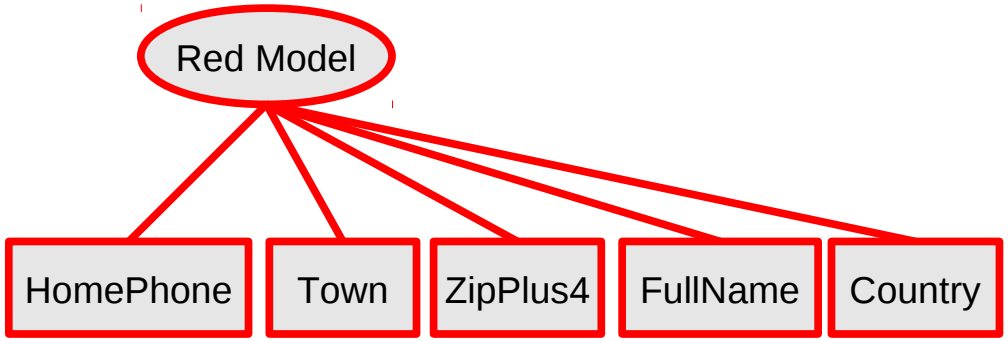
Multi-schema friendly

- Blue App has model



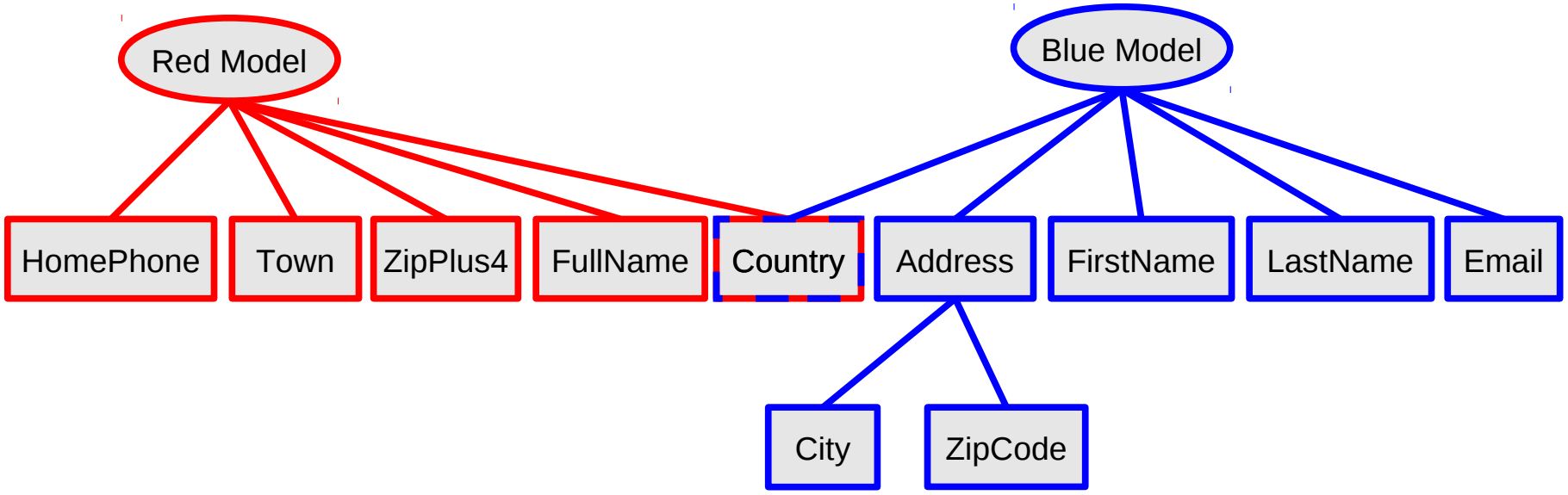
Multi-schema friendly

- Red App has model



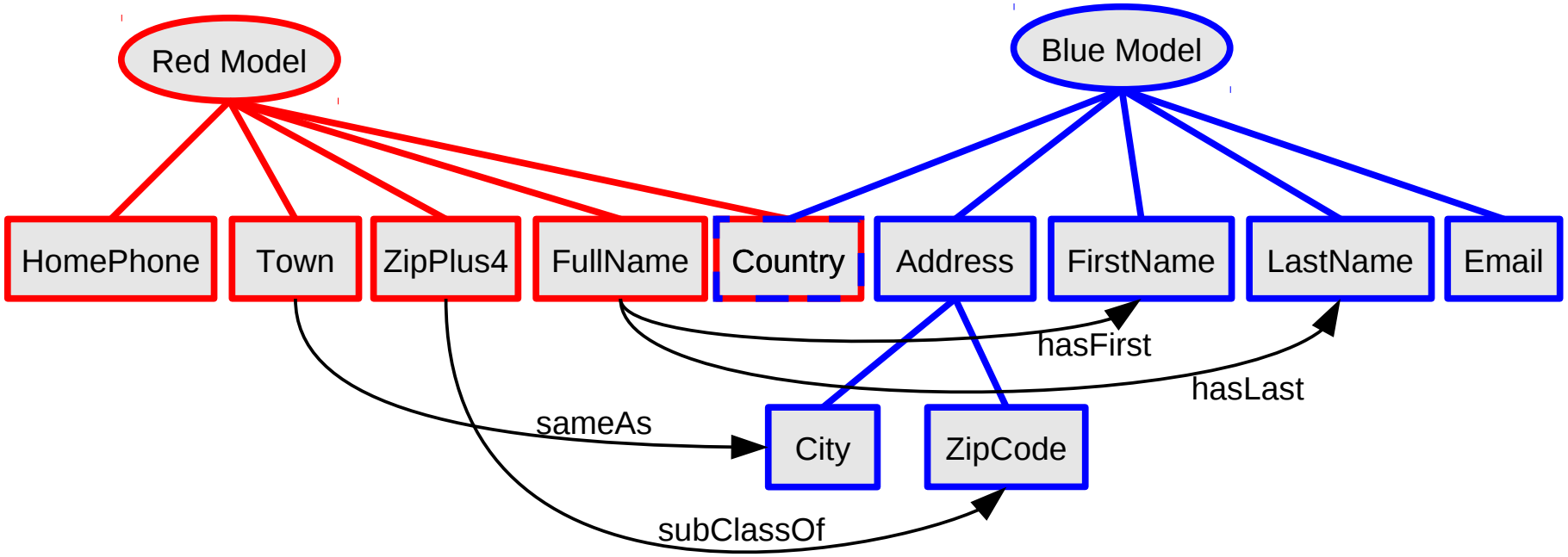
Multi-schema friendly

- Merge RDF data
- Same nodes (URIs) join automatically



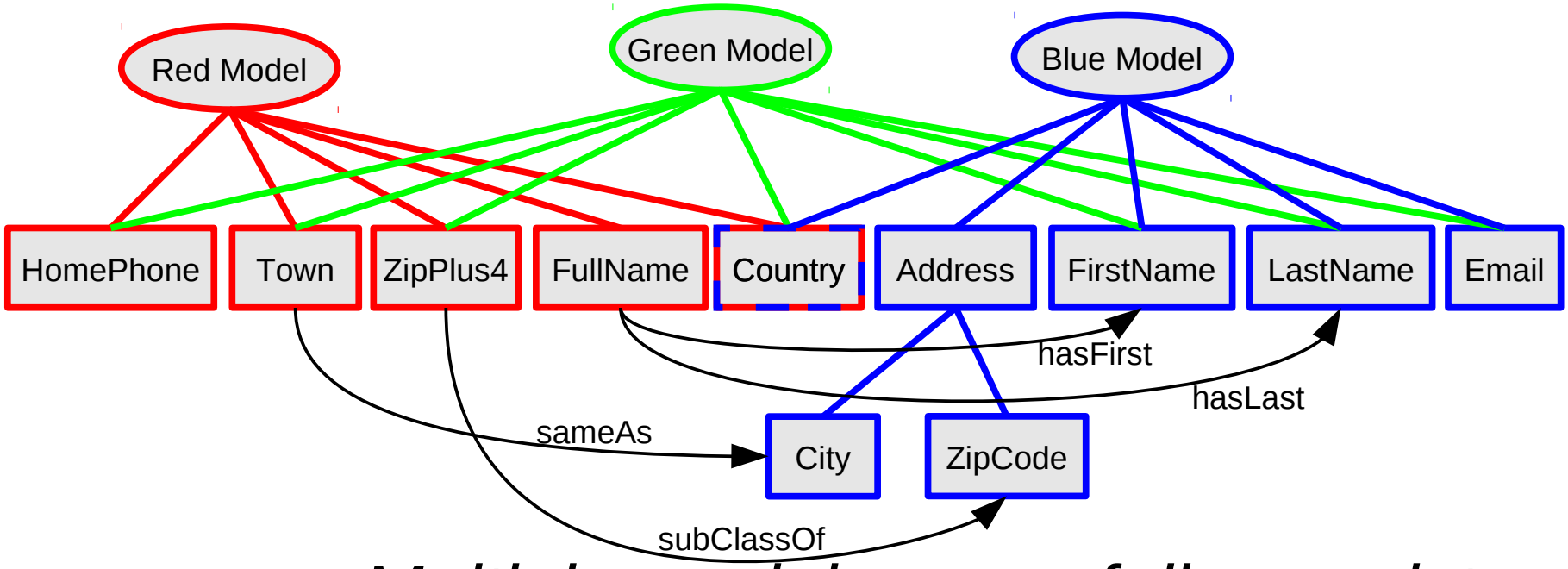
Multi-schema friendly

- Add relationships and rules
- (Relationships are also RDF)



Multi-schema friendly

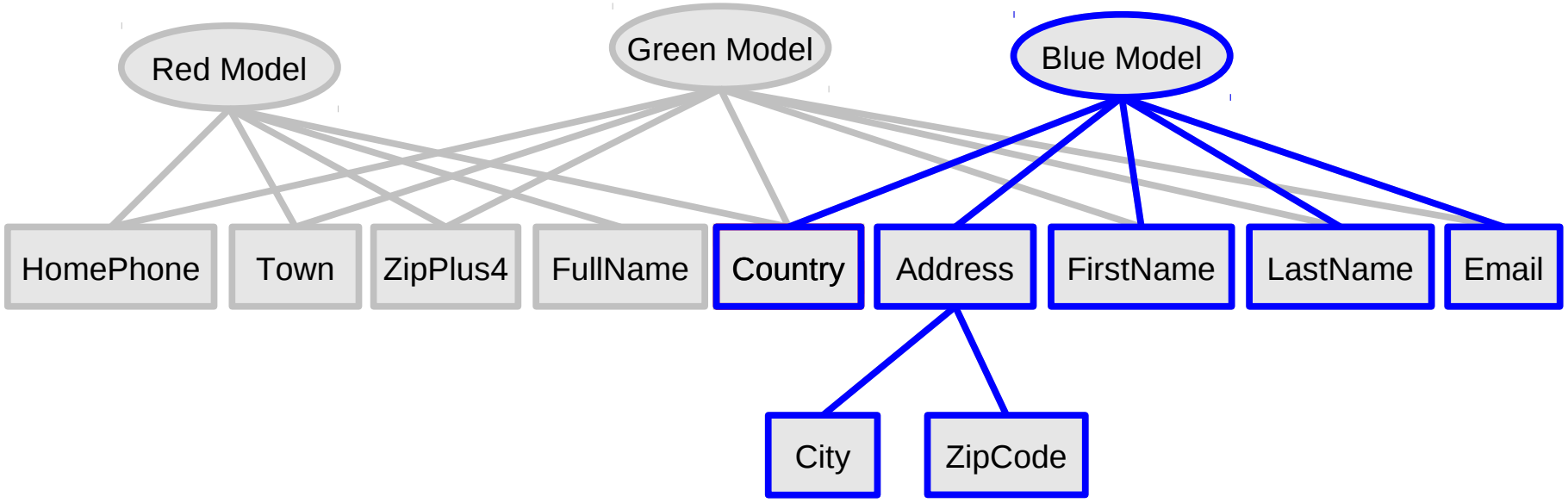
- Later add Green model
  (Using Red & Blue models)



Multiple models peacefully coexist

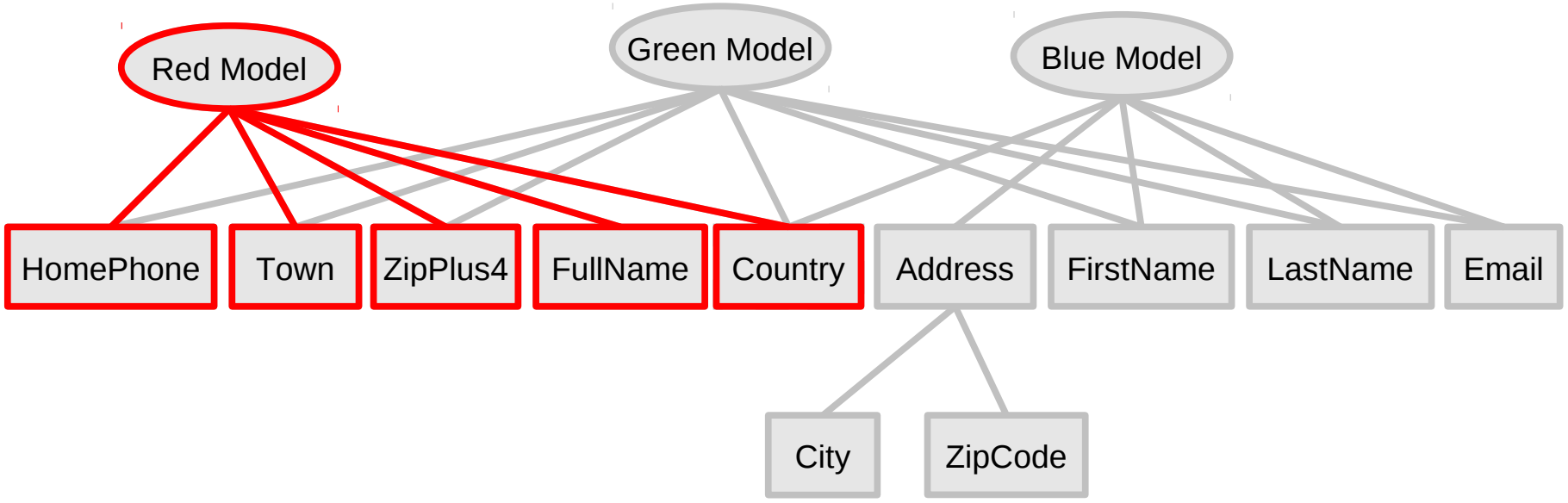
Multi-schema friendly

- What the Blue app sees:
  - No difference!



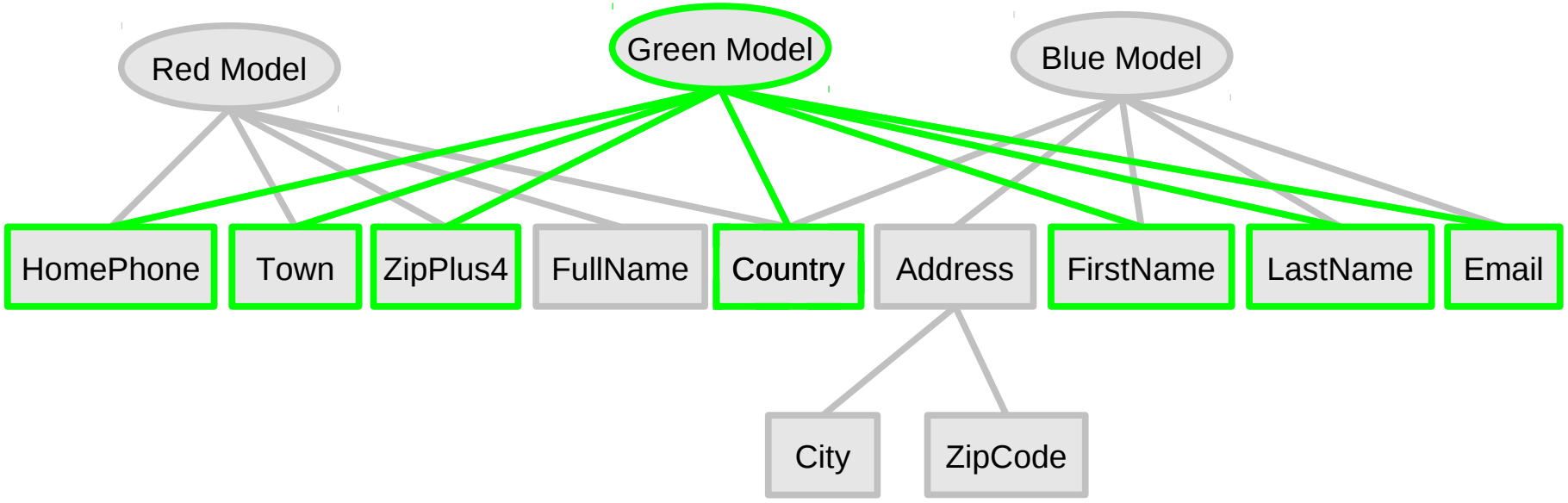
Multi-schema friendly

- What the Red app sees
  - No difference!



Multi-schema friendly

- What the Green app sees
  - No difference!



Why is this important?

- Multiple data models and vocabularies can be:
  - added dynamically
  - used together harmoniously
- This is critical in domains that involve many or changing data models/vocabularies
  - E.g., standard + non-standard models/vocabularies
- Even standards are not static!
  - Standards are continually revised or they become obsolete

Unified Medical Language System (UMLS) includes over 100 standard vocabularies and millions of concepts!

Easy to **merge** data?

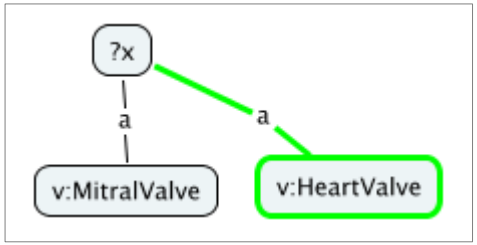
- XML:
  - Schemas compete to be "on top"
  - Meaningful merge requires new schema and manual mapping

- JSON:
  - A little easier than with XML
  - But meaningful merge still requires new model and manual mapping

#1: RDF enables smarter data use and automated data translation

- RDF enables inference
- Inference derives new assertions from old
  - "Entailments"

Inference example

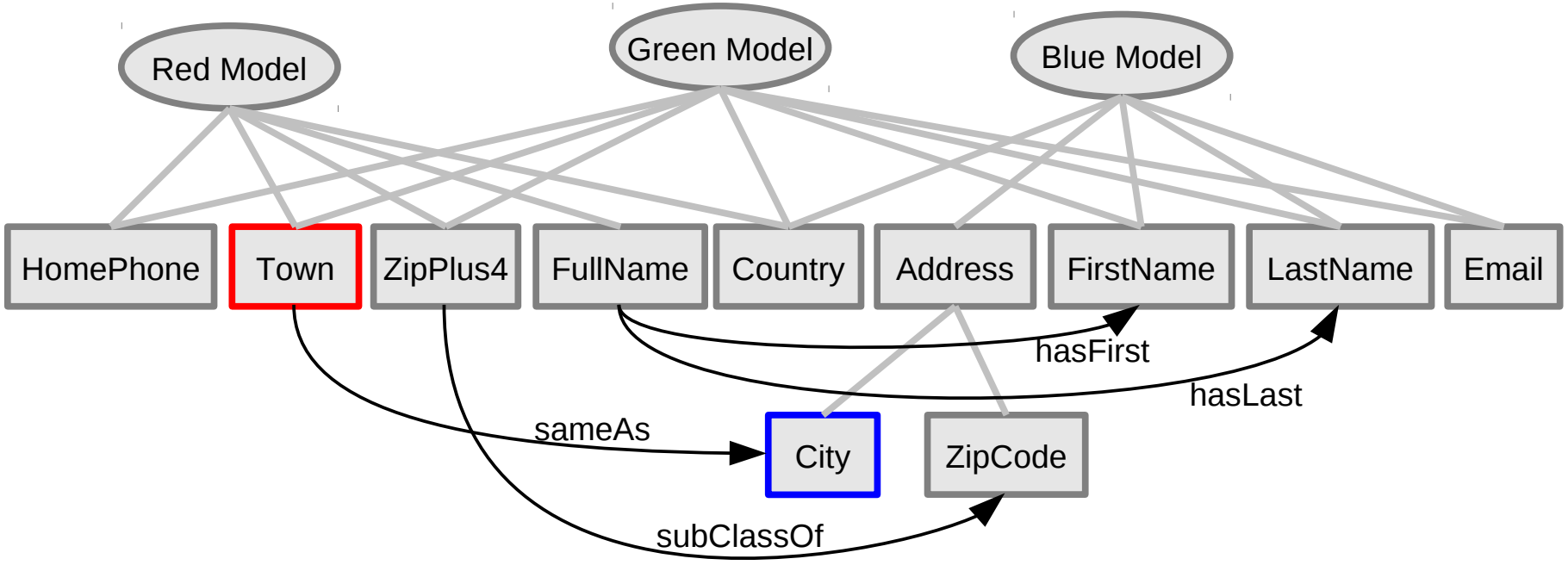


- If you know:
  ?x a v:MitralValve .
  v:MitralValve rdfs:subClassOf v:HeartValve .
- Then you can infer:
  ?x a v:HeartValve .

Why is this important?

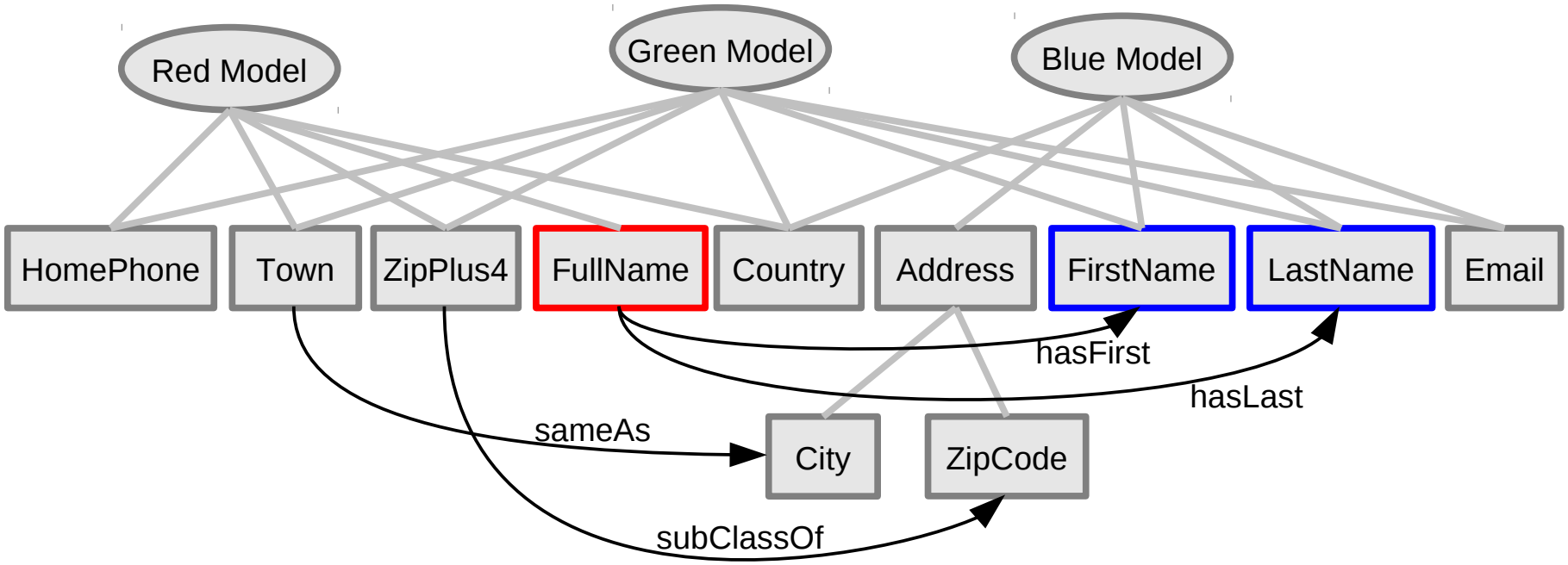
- Smarter queries and data use
  - Query for v:HeartValve surgeries can find v:MitralValve surgeries

Inference example: sameAs



- If you know: Town
- You can infer: City (or vice versa)

Inference example: composition



- If you know: FirstName + LastName
- You can infer: FullName
  - But not necessarily vice versa

Facilitates smarter queries?

- XML:
  - No inference

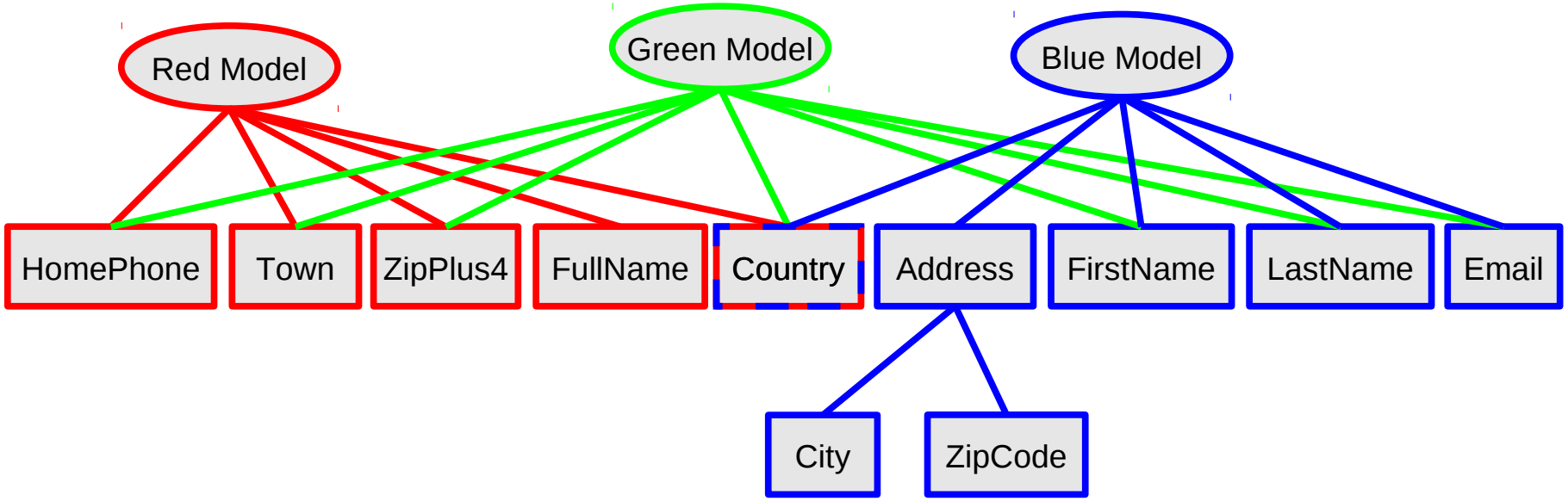
- JSON:
  - No inference

Why is this important?

- Data can be automatically **transformed** between different data models and vocabularies
  - E.g., db:DB00945 => v:aspirin
  - Red Model data + Blue Model data => Green Model data

Very helpful for data integration!

Inference example: data transformation



- If you know: Red Model data + Blue Model data
- You can infer: Green Model data

Facilitates data transformations?

- XML:
  - Not by inference, but tools are available

½

- JSON:
  - Not by inference, but tools are available

½

Weaknesses of RDF

- RDF tools are less mature; expertise is less widespread
- RDF has some annoyances:
  - "Blank nodes" have subtleties that add complication (Best to limit their use)
  - URI allocation – can be a hassle
- Weaknesses should be understood, but are not show stoppers

Conclusions

- RDF provides key benefits that distinguish it from other frequently used information representations
- RDF is best for problems that involve:
  - Large-scale information integration
  - Semantically connecting diverse vocabularies and data models
  - Changing vocabularies and data models
  - Inference and information transformation

Questions?

BACKUP SLIDES

Key things you need to know about RDF

#5: RDF is self describing

- RDF uses URIs as identifiers

#4: RDF is easy to map from other data representations

- RDF data is made of assertions

#3: RDF captures information – not syntax

- RDF is format independent

#2: Multiple data models and vocabularies can be easily combined and interrelated

- RDF is multi-schema friendly

#1: RDF enables smarter queries and automated data translation

- RDF enables inference