FHIR in JSON-LD?

David Booth, PhD
Yosemite Project

HL7 ITS / W3C HCLS
RDF for Semantic Interoperability Group

18-Mar-2015

Problem Statement

- **Q:** Could we retrofit FHIR's JSON syntax to also be JSON-LD?

Update 19-Mar-2019

- For a brief time (2016), the FHIR build process was modified to generate JSON-LD 1.0 that was different JSON than the plain FHIR/JSON format. See:
  - http://dbooth.org/2015/fhir/json-ld/fhir-in-json-ld-OLD.pdf
- The existence of two different JSON formats (plain JSON vs JSON-LD) turned out to be too confusing to users, so the JSON-LD format was subsequently dropped.
- However, W3C is now working on JSON-LD 1.1, which adds features for nested properties within an @context. See an example:
  - https://tinyurl.com/y3fm733s
- We are now exploring whether regular FHIR/JSON could be augmented by a JSON-LD 1.1 @context having nested properties, to enable FHIR to be used as RDF.

OLD SLIDES

18-Mar-2015

Problem Statement

- **Q:** Could we retrofit FHIR's JSON syntax to also be JSON-LD?
- **Short answer:** No. It would require some changes to FHIR/JSON that would be unpalatable to some plain JSON users.

Detailed explanation follows . . .

Background

- JSON-LD is a JSON format that is **also** directly interpretable as RDF
  - Supports inference
  - Supports large-scale information integration
  - Support semantic interoperability
  - See:
    - Why RDF for Healthcare?
    - JSON-LD Articles and Presentations
- We are adding an RDF format to FHIR
- FHIR already offers a (plain) JSON format
- Could we "kill two birds with one stone" by modifying FHIR/JSON to become FHIR/JSON-LD?
- These slides examine that possibility . . .

Observation: Direct transliteration

- Best to use direct transliteration – KISS principle
  - JSON and Turtle would look very similar
- Making URIs out of IDs would cause problems:
  - `http://example/fhir/Observation/12345 <=> "id": "12345"`
  - Ensuring valid URIs when concatenating base URI + ID
  - Round-tripping: parsing IDs back out of the URIs
- Better to use fhir:resourceType:
  - `base:obs123 fhir:resourceType fhir:Observation .`
  - `fhir:resourceType rdfs:subPropertyOf rdf:type . # From ontology`
- instead of:
  - `base:obs123 rdf:type fhir:Observation .`

Round-tripped JSON-LD - dob

```
1. {
2.   "@context": {
3.     "fhir": "http://example/fhir#",
4.     "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
5.     "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
6.     "xsd": "http://www.w3.org/2001/XMLSchema#"
7.   },
8.   "@graph": [
9.     {
10.      "@id": "_:fc7725329340449efa72b6f7f5d7182eeb3",
11.      "http://example/fhir/vocab#url": "http://example.org/fhir/extensions#text",
12.      "http://example/fhir/vocab#valueString": "Easter 1970"
13.    },
14.    {
15.      "@id": "_:fc7725329340449efa72b6f7f5d7182eeb2",
16.      "http://example/fhir/vocab#extension": {
17.        "@id": "_:fc7725329340449efa72b6f7f5d7182eeb3"
18.      },
19.      "http://example/fhir/vocab#id": "314159"
20.    },
21.    {
22.      "@id": "_:fc7725329340449efa72b6f7f5d7182eeb1",
23.      "fhir:_dob": {
24.        "@id": "_:fc7725329340449efa72b6f7f5d7182eeb2"
25.      },
26.      "fhir:dob": {
27.        "@type": "xsd:date",
28.        "@value": "1972-11-30"
29.      }
30.    }
31.  ]
32. }
```

@context embedded

@graph wrapper

Prefixes are lost

RDF blank node labels change

CONCLUSION: Generic RDF-->JSON-LD re-serializers will not suffice

Good: Most issues seem manageable

- Blank nodes
- Extensions can be related:

```
fhir:dob fhir:extension fhir:_dob
```

- FHIR RDF will need custom serialization to JSON-LD anyway

Good: modifier extensions okay

- FHIR RDF can capture raw message structure, with no semantics
- To add semantics, define FullyUnderstood subclass, e.g.:
  - fhir:RawObservation (superclass)
    - fhir:FullyUnderstoodObservation

Good: FHIR has a closed content model

- Extensions cannot define new elements/attributes
- This simplifies JSON<=>RDF mapping
  - @context can fully enumerate the JSON/RDF properties

Bad: Element type depends on context

```
"resourceType": "Observation",
"id": "example",
"code": {
  "coding": [
    {
      "system": "http://loinc.org",
      "code": "3141-9",
      "display": "Weight Measured"
    }
  ]
},
```

Object

String

RDF lists

- JSON-LD will generate RDF lists to preserve ordering – required by FHIR
- RDF lists are not SPARQL or inference friendly
  - But RDF->RDF translation will likely be needed anyway to best support inference

Potential @context for "code"

```
"@context":
{
   "@base": "http://example/base/",
   "base":  "http://example/base/",
   "@vocab": "http://example/fhir#",
   "fhir":   "http://example/fhir#",
   "resourceType": { "@type": "@vocab" },
   "code": {
     "@id": "fhir:code",
     "@container": "@list"
   },
   "Observation": {"@id": "fhir:Observation"}
},
```

JSON-LD versus Turtle

```
{ "@context": ...
  "resourceType": "Observation",
  "code": {
    "coding": [
      {
        "system": "http://loinc.org",
        "code": "3141-9"
      }
    ]
  },
  "foo": {
    "code": [ "aaa", "bbb" ]
  },
  "bar": {
    "code": [ { "ccc": "ddd" } ]
  }
}
```

```
[] fhir:resourceType
            fhir:Observation ;
   fhir:code ( [
      fhir:coding
      [
        fhir:system "http://loinc.org" ;
        fhir:code ( "3141-9" )
      ]
   ] ) ;
   fhir:foo [
      fhir:code ( "aaa" "bbb" )
   ] ;
   fhir:bar [
     fhir:code ([fhir:ccc "ddd"])
   ] .
```

Potential solutions

- Option 1: Add @context to every nested JSON element that may require different element mapping
- Option 2: Wrap all values in RDF lists

```
[] fhir:code ( "3141-9" ) .
```

- Option 3: ??? (Asking JSON-LD experts)

Options

- JSON-LD:
  - Avoids a third FHIR serialization
  - W3C standard
  - Can standardize for FHIR very quickly
- Custom "ideal" RDF:
  - Can be better for inference
  - Will take much longer to standardize
- Combined approach: Adopt direct RDF now (either JSON-LD or custom RDF) + "ideal" RDF later
  - Open question: How much gap is there between direct RDF and "ideal" RDF? Could direct RDF be good enough?

EVEN OLDER SLIDES -- OBSOLETE

David Booth, PhD
Yosemite Project

HL7 ITS / W3C HCLS
RDF for Semantic Interoperability Group

24-Feb-2015

Outline

- Observations & Conclusions
- Background & Motivation
- Issues in adopting JSON-LD for FHIR

Observations

- Retrofitting FHIR JSON to use JSON-LD will not work without changing FHIR JSON
  - Information loss when interpreted as RDF
- FHIR JSON partially reinvents JSON-LD
  - Types
  - Identifiers
- FHIR JSON seems to be designed as a second-class citizen to mimic capabilities of FHIR XML

Conclusions

- For JSON-LD to work well with FHIR, some FHIR details would probably have to be changed
  - E.g., identifier scoping and some JSON elements
- Benefits of FHIR using JSON-LD:
  - It's already RDF (no additional mapping, global clarity, etc.)
  - Based on existing W3C JSON-LD standard, rather than reinventing
- Good option to pursue, but needs more active participants
  - Enough interest?
  - Timely enough?

What is JSON-LD?

- Both:
  - A form of JSON
  - An RDF serialization

Good starting point: http://json-ld.org/

Why JSON-LD for FHIR?

- JSON-LD was designed to enable many JSON documents to be retrofitted to become JSON-LD
  - By addition of a @context tag
- FHIR already has a JSON serialization

Could JSON-LD allow FHIR to be <u>both</u> JSON and RDF?

JSON-LD Example:
lennon.jsonld

```
1.{
2.  "@context": "http://dbooth.org/2015/fhir/json-
   ld/person.jsonld",
3.  "@id": "http://dbpedia.org/resource/John_Lennon",
4.  "name": "John Lennon",
5.  "born": "1940-10-09",
6.  "spouse":
   "http://dbpedia.org/resource/Cynthia_Lennon"
7.}
```

JSON-LD context: http://dbooth.org/.../person.jsonld

```
1.{
2.   "@context":
3.   {
4.      "Person": "http://xmlns.com/foaf/0.1/Person",
5.      "xsd": "http://www.w3.org/2001/XMLSchema#",
6.      "name": "http://xmlns.com/foaf/0.1/name",
7.      "born":
8.      {
9.         "@id": "http://schema.org/birthDate",
10.        "@type": "xsd:date"
11.     },
12.     "spouse":
13.     {
14.        "@id": "http://schema.org/spouse",
15.        "@type": "@id"
16.     }
17. }
```

JSON-LD Example

JSON-LD
lennon.jsonld

```
{
  "@context": "http://dbooth.org/2015/fhir/json-ld/person.jsonld",
  "@id": "http://dbpedia.org/resource/John_Lennon",
  "name": "John Lennon",
  "born": "1940-10-09",
  "spouse": "http://dbpedia.org/resource/Cynthia_Lennon"
}
```

Context
http://dbooth.org/.../person.jsonld

```
{
  "@context":
  {
    "Person": "http://xmlns.com/foaf/0.1/Person",
    "xsd": "http://www.w3.org/2001/XMLSchema#",
    "name": "http://xmlns.com/foaf/0.1/name",
    "born":
    {
      "@id": "http://schema.org/birthDate",
      "@type": "xsd:date"
    },
    "spouse":
    {
      "@id": "http://schema.org/spouse",
      "@type": "@id"
    }
  }
```

For FHIR in JSON-LD

- Need to define @context for JSON terms
- May want to tweak FHIR JSON to better align with JSON-LD style
  - Officially adopt JSON-LD style?

FHIR XML – dob (primitive element with extension)

```
<dob id="314159" value="1970-03-30" >
  <extension url="http://example.org/fhir/extensions#text">
    <valueString value="Easter 1970"/>
  </extension>
</dob>
```

See http://www.hl7.org/implement/standards/fhir/json.html

FHIR JSON – dob

In FHIR JSON it becomes two properties:

```
{
"dob": "1972-11-30",
 "_dob": {
   "id": "314159",
   "extension": [{
      "url" : "http://example.org/fhir/extensions#text",
      "valueString" : "Easter 1970"
   }]
 }
```

FHIR JSON-LD – dob

```
{
 "@context": "http://dbooth.org/2015/fhir/json-ld/dob-context.jsonld",
 "dob": "1972-11-30",
 "_dob": {
   "id": "314159",
   "extension": [{
      "url" : "http://example.org/fhir/extensions#text",
      "valueString" : "Easter 1970"
   }]
 }
}
```

FHIR JSON-LD – dob @context

```
1.{
2.   "@context":
3.   {
4.      "@vocab": "http://example/fhir/vocab#",
5.      "fhir": "http://example/fhir#",
6.      "xsd": "http://www.w3.org/2001/XMLSchema#",
7.      "dob":
8.      {
9.         "@id": "fhir:dob",
10.         "@type": "xsd:date"
11.      },
12.      "_dob":
13.      {
14.         "@id": "fhir:_dob",
15.         "@type": "@id"
16.      }
17.   }
18.}
```

Primitive extension names

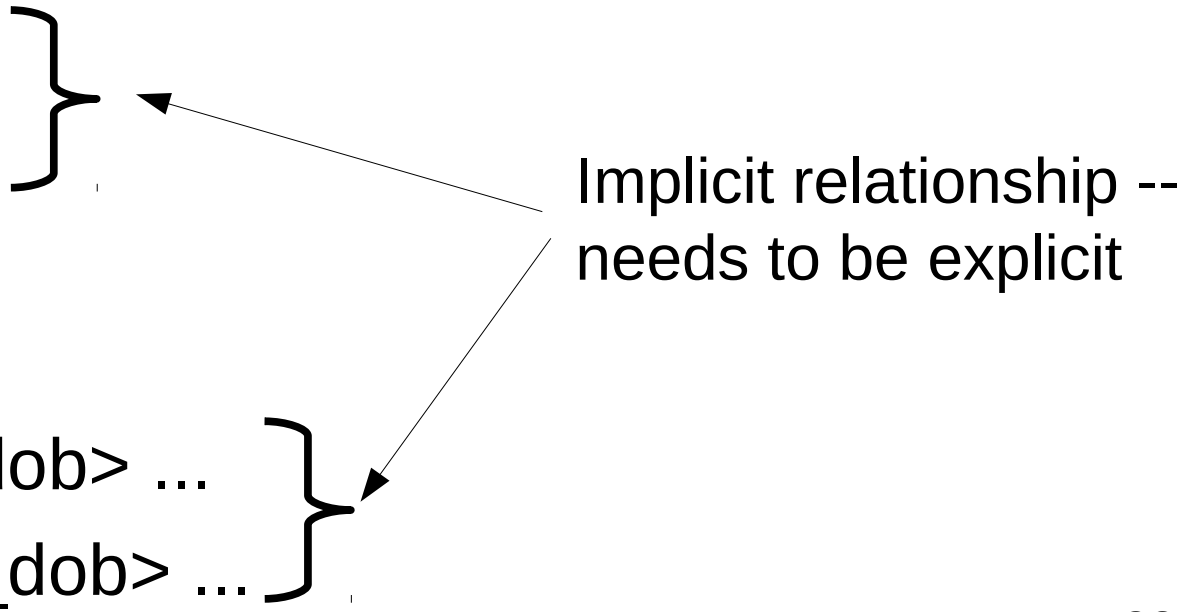
XML:

<dob ...>

JSON:

"dob": ...

"_dob": ...



RDF:

<http://...#dob> ...

<http://...#_dob> ...

RDF / Turtle - dob

Implicit relationship between fhir:dob and fhir:_dob needs to be explicit:

```
 1. @prefix fhir:  <http://example/fhir#> .
 2. @prefix xsd:  <http://www.w3.org/2001/XMLSchema#> .
 3.
 4. _:b0 fhir:_dob  _:b1 ;
 5.      fhir:dob   "1972-11-30"^^xsd:date .
 6.
 7. #  fhir:_dob fhir:extends fhir:dob .     # From ontology?
 8.
 9. _:b1 <http://example/fhir/vocab#extension>  _:b2 ;
10.      <http://example/fhir/vocab#id>  "314159" .
11.
12. _:b2 <http://example/fhir/vocab#url>
13.          "http://example.org/fhir/extensions#text" ;
14.      <http://example/fhir/vocab#valueString>
15.          "Easter 1970" .
```

Issue 1: Extensions are implicitly related

`fhir:dob` vs. `fhir:_dob`

- Solution A: Ontology could relate them, but this would mean that round-tripping would depend on the ontology. Maybe okay for extensions to standard elements, but not for extensions to extensions.
- Solution B: Add explicit statement to JSON:

```
"dob":
  {
     "@id": "fhir:dob",
     "@type": "xsd:date",
     "extension": "fhir:_dob"
  }
```

Round-tripped JSON-LD - dob

```
1. {
2.   "@context": {
3.     "fhir": "http://example/fhir#",
4.     "rdf": "http://www.w3.org/1999/02/22-rdf-syntax-ns#",
5.     "rdfs": "http://www.w3.org/2000/01/rdf-schema#",
6.     "xsd": "http://www.w3.org/2001/XMLSchema#"
7.   },
8.   "@graph": [
9.     {
10.      "@id": "_:fc7725329340449efa72b6f7f5d7182eeb3",
11.      "http://example/fhir/vocab#url": "http://example.org/fhir/extensions#text",
12.      "http://example/fhir/vocab#valueString": "Easter 1970"
13.    },
14.    {
15.      "@id": "_:fc7725329340449efa72b6f7f5d7182eeb2",
16.      "http://example/fhir/vocab#extension": {
17.        "@id": "_:fc7725329340449efa72b6f7f5d7182eeb3"
18.      },
19.      "http://example/fhir/vocab#id": "314159"
20.    },
21.    {
22.      "@id": "_:fc7725329340449efa72b6f7f5d7182eeb1",
23.      "fhir:_dob": {
24.        "@id": "_:fc7725329340449efa72b6f7f5d7182eeb2"
25.      },
26.      "fhir:dob": {
27.        "@type": "xsd:date",
28.        "@value": "1972-11-30"
29.      }
30.    }
31.  ]
32. }
```

@context embedded

@graph wrapper

Prefixes are lost

RDF blank node labels change

CONCLUSION: Generic RDF-->JSON-LD re-serializers will not suffice

Issue 2: Context embedded

```
"@context": { ... }
```

- Solution: Use a custom FHIR JSON-LD serializer
  - Not a problem. FHIR JSON would need one anyway

Issue 3: @graph wrapper

```
"@graph": { ... }
```

- Solution A: Use a custom FHIR JSON-LD serializer
- Solution B: Allow it in the FHIR JSON-LD syntax

Issue 4: Prefixes are lost

"url" (JSON) -->

<http://example/fhir/vocab#url> (RDF) -->

"http://example/fhir/vocab#url" (JSON)

- Solution A: Add explicit prefix information:

```
"prefix": [ "fhir", "http://example/fhir/vocab#" ]
```

- Solution B: Add to ontology, either:

```
<http://example/fhir/vocab#url> fhir:localName "url" .
<http://example/fhir/vocab#> fhir:prefix "fhir" .
```

Issue 5: Blank node labels change

```
"@id": "_:fc7725329340449efa72b6f7f5d7182eeb3"
```

- Why:
  - JSON objects do not always have explicit @ids
- Solution: Require explicit @ids.
  - Option A1: Require them in XML also.
  - Option A2: Define a standard way to auto-name @ids, based on XML nesting structure (XPath).

JSON-LD round tripping conclusions

- FHIR-specific JSON / JSON-LD serializers will be required
- RDF will have to retain some serialization artifacts:
  - Prefix definitions
  - Blank node labels? (If JSON-LD is used with @id)

JSON-LD alignment

FHIR JSON partially re-invents JSON-LD:

FHIR JSON	JSON-LD
"resourceType": "Procedure"	"@type": "Procedure"
"id": "org1" (but scope is resource)	"@id": "org1" (scope is global)
Coding system URL	URL in @context definition
Extension "url"	@type

TO DO: Investigate modifier extensions

- Option A: Have FHIR RDF capture FHIR message structure, with no semantics, then add semantics later through separate ontologies, perhaps after RDF->RDF translation
- Option B: Define resource superclass with Modified and Unmodified subclasses, e.g.:
  - fhir:Observation (superclass)
    - fhir:ObservationUnmodified
    - fhir:ObservationModified

TO DO: Investigate identifiers further

```
<dob id="314159" value="1970-03-30" >
```

- What approaches could be used to ensure globally unique identifiers from FHIR XML?
  - Individual resources & bundles (Atom feed)

Summary of required FHIR JSON changes

1. Add @context to every FHIR JSON instance document, or link externally, such as by an HTTP Link header.
2. Change "id" to "@id" in JSON. (TODO: Figure out how to handle the scoping properly.)
3. Always permit an @id in every element. (TODO: File FHIR change request regarding profiles)
4. Change extension "url" to "@type". (TODO: Verify that "url" is not used elsewhere, or this context can be distinguished.)

Conclusions

- For JSON-LD to work well with FHIR, some FHIR details would probably have to be changed
  - E.g., identifier scoping and some JSON elements
- Benefits of FHIR using JSON-LD:
  - It's already RDF (no additional mapping, global clarity, etc.)
  - Based on existing W3C JSON-LD standard, rather than reinventing
- Good option to pursue

QUESTIONS?

Other Notes (mostly to self)

JSON-LD uses CURIEs

- Element should not contain a colon:
  <foo:bar> <foo:bar:baz>

Resource identity

http://www.hl7.org/implement/standards/fhir/managing.html

- Each resource has a URL (outside the resource)
- Many resources also have a logical identifier, which is different from the literal identity of the resource. Multiple resource instances can describe the same concept across multiple systems.
- "The full identity of a resource is an absolute URL constructed from the server address at which it is found, the resource type, and the Logical Id, such as http://test.fhir.org/rest/Patient/123 (where 123 is the Logical Id)." -- http://www.hl7.org/implement/standards/fhir/resources.html
  - SERVER_URI/TYPE/LOGICAL_ID

FHIR Resources

- http://www.hl7.org/implement/standards/fhir/overview.html
- Resources all have:
  - Common way to define and represent them, building from data types
  - A common set of metadata
  - A human readable part

Special Resources

- Conformance Statement – describes an implementation's interfaces
- Profile – constrains optionality, cardinality, terminology bindings, data types and extensions

Resource definition

- http://www.hl7.org/implement/standards/fhir/resources.html
- A resource is an entity that:
- has a known identity (a url) by which it can be addressed
- identifies itself as one of the types of resource defined in this specification
- contains a set of structured data items as described by the definition of the resource type
- contains a human-readable XHTML representation of the content of the resource
- has an identified version that changes if the contents of the resource change
-

Contents of a Resource

- http://www.hl7.org/implement/standards/fhir/resources.html#content
- Resources have:
- A base set of defined data elements specific to the type
- Extensions - additional data elements added by implementations
- A human-readable narrative description of the contents of the resource
- Contained resources - additional resources that are part of the identification and transaction space of this resource
- Metadata - important information about the resource that is not part of the content model of the resource:
  - Logical Id
  - Versioin Id
  - Last Modified Date
- Tags - labels affixed to the resources that may be used to define additional operational behavior such as security, workflow, etc.
-

Issue: Minting URIs from identifiers

- We need a way to reliably generate a new URI (for RDF) from a URI+localName
- For this to work well in JSON-LD, it should be concatenation. But this means that the syntax of the original URI must be constrained, to ensure that concatenation will work, i.e., be syntactically legal and not clash with any other URIs

Message Identifiers

- http://www.hl7.org/implement/standards/fhir/messaging.html
- Envelope ID (feed.id), changed each time a message is sent
- Message ID (unique within the message stream)
- Q: How is the message stream identified?
-

Issue: Language element

- http://www.hl7.org/implement/standards/fhir/resources.html
- "specifies the base language of the resource"
- Should this be propagated to RDF lang tags?
- Prob not. It looks like it will be used for narratives, which are specified in xhtml, which has lang attributes:
  - http://www.w3.org/International/questions/qa-html-language-declarations

Issue: Lists

- Q: Can new extension elements contain lists? If so, how will the JSON parser know that?
- Q: Normally a JSON-LD list produces an unordered **set** of triples, which seems like a loss of information. If there is only one member of the set, how would a JSON-LD serializer (from RDF) know that it should be serialized as a list rather than as a single property?
- A: Not a problem. FHIR is a closed content model, so no new elements are possible. Therefore, all lists can be declared in the @context.