JSON Representation of DICOM Structured Reports

DICOM WG 23
David Clunie
Presentation of Public Comment Draft to WG 6
2019/11/07
AI CHANGES THE GAME

http://medium.com/adhive/disruptive-ai-controlled-advertising-cd90a07452cb
Annotation interoperability matters now

• Previously:
  • little incentive to annotate
  • few tools to create or view annotations
  • annotation interoperability was a low priority for product managers
  • presentation rather than semantics were the priority for annotation tools

• Now:
  • semantic annotations have (real monetary) value beyond primary use case
  • recognition of existence of unanticipated re-use cases
  • annotations are expensive to create/recreate retrospectively
  • more expensive to process if proprietary rather than OTS standard
  • AI-generated annotations need to be interoperable for display
  • “interactive” AI requires interoperable annotation exchange
  • AI vendors unlikely to be the same as scanner/PACS vendors – mix and match
DICOM SR and AI

• DICOM SR is a generic solution for:
  • fundamental encoding of measurements, categorical results, using codes and referencing images, waveforms as well as spatial and temporal coordinates
  • reusable sub-templates for specific scenarios that are common to different use cases and applications
  • generic root level templates for non-specific measurements (e.g., TID 1500)
  • linking other objects related to results and measurements (such as SEG, Parametric Map and RWVM)

• Specific templates for:
  • traditional CAD applications that are relevant to AI
  • traditional human operator measurements that may now be made by AI
DICOM SR and the developer

- Traditional DICOM SR encoding requires use of a toolkit and an API with a non-trivial learning curve (binary encoding intractable by hand)
- AI algorithm developer may not need to know about the “composite context” (patient/study/series +/- workflow metadata) of the encounter
- Impedance mismatch between
  - PACS-orientated “DICOM image in, DICOM SEG + SR out”
  - Algorithm-developer orientated “PNG in, PNG + JSON out”
- Even XML is deemed excessive/too complicated by AI developer community
- DICOMweb JSON encoding is also intractable for SR, since it is hexadecimal tag, individual data element orientated (no SR content item abstraction)
Goals for Simplified DICOM SR in JSON

• Full-fidelity round trip with actual DICOM SR for all constructs (any template)
• Simple (enough to hand write or copy from examples)
• Compact (even terse)
• Understandable (relatively)
• Unambiguous (easily parsable)
• Leverage any existing actual or de facto JSON or evolving AI standards
• Platform independent
• Capable of encoding extracts separated from composite context (such as without “header” rather than content tree, image library, etc., which could be added by separate tool/pass)
Non-Goals for Simplified DICOM SR in JSON

• Not an alternative/competitor to existing PS3.18 Annex F JSON for non-SR objects
• Not an alternative/competing persistent form to be serialized and stored, as opposed to binary DICOM SR stored in PACS/VNA
• Not an abstraction of template-specific concepts or alternative information models for similar content
• No template-specific constraints or optimizations
• Not a means for defining a new validation mechanism for SR content (template-defined), but does not prohibit it
Pipeline to add missing stuff to JSON

Just Number, Coordinates and Image Reference

+ Lesion Identifiers

+ Image Library and Evidence Sequences

+ Composite Context

AI Algorithm

Lesion Manager

DICOM Image Aware System

Patient-Study Aware System

PACS
Design Decisions – Business Names

• No hexadecimal numbers for “header” attributes – leverage DICOMweb JSON encoding but with PS3.6 keywords rather than numeric tags
• Abstract the content items (i.e., name-value pairs), as if they were attributes, rather than exposing their component attributes
• No obscure alphanumeric codes in content tree – use “business names” concept from Green CDA (not dissimilar to JSON-LD)
• Codes are defined in separate “business names” JSON file that acts as a dictionary – do not need to be standardized (but may be in future, like keywords)
Design Decisions – JSON Structure

• Use JSON Objects where identity is important but not order
• Use business name as name of JSON Object’s name-value pair
• Use JSON Arrays to preserve order
• Use JSON Arrays to allow sibling JSON Objects with same name
• Use a JSON Array to encode children
• Collapse unnecessary JSON Arrays into single value when possible for business names and top level data elements
• Omit data element VR if it can be found in dictionary or business name file
• Omit explicit value type and relationship type if they can be deduced from context, or defined in the separate business names file
• Add annotations (specific object names starting with “@” symbol) to resolve ambiguities, and to provide target and source for by-reference relationships
Example 1 – dcsrump of the original
Example 1 – JSON of the content tree (only)

```
"ImagingMeasurementReport": [
  {
    "@tm": "DMR",
    "@tid": "1500"
  },
  {
    "ImagingMeasurements": [
      {
        "MeasurementGroup": [
          {
            "TrackingIdentifier": "5b6eb4301d31759a2d29985a3d8fb00"
          },
          {
            "TrackingUniqueId": "1.3.6.1.4.1.5962.1.1.0.0.0.1535644357.22655.1"
          },
          {
            "Length": [
              "66.43856134",
              "mm",
              [
                [
                  "POLYLINE",
                  172.835376601562,
                  270.064069148625,
                  133.79888916015625,
                  343.0453186035156
                ],
                [
                  "1.2.840.10008.5.1.4.1.1.2",
                  "1.3.6.1.4.1.14519.5.2.1.9203.4000.26801842288818573226516023762"
                ]
              ]
            }
          }
        ]
      }
    ]
  }
]```
"ImagingMeasurements": [
  {
    "MeasurementGroup": [
      {
        "TrackingIdentifier": "5b6eb4301d3175942d29985a3d0fbb00"
      },
      {
        "TrackingUniqueIdentifier": "1.3.6.1.4.1.5962.1.1.0.0.1535644357.22655.1"
      },
      {
        "Length": [
          "66.43856134",
          "mm",
          [
            "POLYLINE",
            [
              172.83535766601562,
              270.0640869140625,
              133.7988916015625,
              343.0453186035156
            ],
            [
              "1.2.840.10008.5.1.4.1.1.2",
              "1.3.6.1.4.1.14519.5.2.1.9203.4004.26801842288818573226516023762"
            ]
          ]
        }
      }
    ]
  }
]
Example 1 – JSON of result only (no id, coord)

```json
"ImagingMeasurementReport": [
  {
    "ImagingMeasurements": [
      {
        "MeasurementGroup": [
          {
            "Length": [
              "66.43856134",
              "mm"
            ]
          }
        ]
      }]
  }
]
```
Example 1 – Business Names file (partial)

```json
{
  "PersonObserverName": {
    "Box": "121886",
    "Boxcd": "SCM",
    "Rcm": "Person Observer Name",
    "Vtt": [
      "VNAME"
    ],
    "Vrel1": [
      "HAS OBS CONTEXT"
    ]
  },
  "CTAbdomen": {
    "Box": "41806-1",
    "Boxcd": "LN",
    "Rcm": "CT Abdomen"
  },
  "StudyDate": {
    "Box": "111886",
    "Boxcd": "CSM",
    "Rcm": "Study Date",
    "Vtt": [
      "DATE"
    ],
    "Vrel1": [
      "HAS ACQ CONTEXT"
    ]
  },
  "FindingSite": {
    "Box": "363698007",
    "Boxcd": "SCT",
    "Rcm": "Finding Site",
    "Vtt": [
      "CODE"
    ],
    "Vrel1": [
      "HAS CONCEPT MOD"
    ]
  }
}
```
{,

"FindingSite": {
    "@cv": "363698007",
    "@csd": "SCT",
    "@cm": "Finding Site",
    "@vt": [
        "CODE"
    ],
    "@rel": [
        "HAS CONCEPT MOD"
    ]
}
}

{,

}
Current Discussion and Open Issues

• Some folks are distressed by the terseness and nesting of objects and arrays

• Proposals for clarity at the expense of compactness – e.g., include annotation to indicate type rather than relying on positional parameters, such as for graphic type, coordinates, etc. – specified as open issue

• Concern about “anonymous” concept names (no name for name-value pair) allowed by both SR infrastructure and many templates, esp. for IMAGE and SCOORD content items – harmless but lead to lots of nesting

• Adding higher level abstractions (than content item level) might address some of these concerns, but also make for a more complicated parser

• Proposal to substitute keywords/business names for UIDs (e.g., SOP Class)
"": ["POLYLINE",
  [172.83535766601562,
   270.0640869140625,
   133.79888916015625,
   343.0453186035156
  ],
  {
    "": [
      "1.2.840.10008.5.1.4.1.1.2",
      "1.3.6.1.4.1.14519.5.2.1.9203.4004.268018422288818573226516023762"
    ]
  }
]

"": {
  "GraphicType": "POLYLINE",
  "value": [
    172.83535766601562,
    270.0640869140625,
    133.79888916015625,
    343.0453186035156
  ],
  "children": {
    "": {
      "ReferencedSOPClassUID": "1.2.840.10008.5.1.4.1.1.2",
      "ReferencedSOPInstanceUID": "1.3.6.1.4.1.14519.5.2.1.9203.4004.268018422288818573226516023762"
    }
  }
}
"SOPClassUID": {
  "Value": [
    "1.2.840.10008.5.1.4.1.1.88.22"
  ]
},

"SOPClassUID": {
  "Value": [
    "EnhancedSRStorageSOPClass"
  ]
}
Out of Scope (for this development cycle)

- A DICOMweb API to transform JSON SR to/from the standard binary DICOM SR persistent form (in a WADO-RS or STOW-RS “application/dicom+json” like manner, e.g., “application/x.dicomsr+json”)
- A DICOMweb API to access, create or modify the DICOM SR content tree abstraction (cf. the existing RetrieveMetadata individual DICOM attribute level access)
- A DICOMweb API to create and manage individual (or sets of) annotations separately from the storage/retrieval of entire DICOM SR object
- A DICOMweb API to perform/manage the various steps of the authoring pipeline that adds lesion management, image references and descriptions, and patient/study/series/workflow composite context