DevDays HL7® FHIR®

HL7 v2 & CDA to FHIR

Grahame Grieve

November 20-22, Amsterdam | @HL7 @FirelyTeam | #fhirdevdays | www.devdays.com

Methods for Transforming content

- Methods (not examples)
- Focus on V2 and CDA because of importance

- Methods:
  - Code
  - Table Mapping Forms
  - Mapping Language
  - Javascript Sandbox

CDA Problem Statement

- Convert Clinical Document to a bundle of resources (or set)
- Output:
  - Composition
  - Patient
  - Practitioner
  - Sections + linked resources

CDA Challenges

- Mapping from Section content to Resource
- Resource Type depends on classCode, moodCode & context
- Or may be indeterminate
- Mapping from statusCode may be incomplete
- GTS → Timing particularly difficult
- Resolution of record identities / snapshot reconciliation may be difficult or impossible

v2 Problem Statement

- Commit a v2 message to a FHIR repository
- E.g perform a transaction
  - Process Message Header
  - Create / Update Patient, Encounter
  - Process other segments

v2 Challenges

- Create vs update appears through the problem (& permutates)
- Update business rules can be significant complex
- |""| - can be very difficult to manage (specially for repeating fields)
- Implicit patterns in OBX sequences can be very difficult to manage
- Many identifiers are local – need to be made global

Basic Rule of thumb

- 90% of elements are simple 1:1 mappings
- 9% of elements involve restructuring / conditional logic
  - E.g. a list of coded elements to a set of named fields (or vice versa)
  - May depend on implicit semantics of coded values
- Last 1% very hard (potentially very hard indeed)
  - Need specific logic to manage restructuring
  - Examples: restructuring implicit OBX trees

Goals

- Do not add to the complexity of the problem
- Be complex enough to meet requirements
  - Depends on how much you're trying to do
- Be fast enough (?)

- Move mapping from being a solitary activity
  - E.g. make mappings portable – do not depend on specific features of implementing system

First principles

- Most data being transformed is a Directed Acyclic Graph (= tree)
  - Root element has an arbitrary name
  - Elements have a primitive value and/or a set of named child Elements
  - Elements have a named type that controls their 'value domain'
  - Elements have an assigned cardinality m..n (0..1, 0..*)
- Structures are different because:
  - Variable use of named properties vs coded repeats
  - Design driven by underlying 'reference model'
  - Original understanding of requirements different
  - Expression of information is fractal

Types of Mappings

- Skeletal Mappings
  - Map the class backbones – pointers about the relationships
- Concept Mappings
  - Map all the elements in the class model
- Detailed Mappings
  - Map all the elements, down to primitive types
- Executable Mappings
  - Describe conversions for all domain values, all special cases

Code

- Everybody's default option: programming language (or XSLT)
  - Transforming data is application #1 for all languages
  - Meets Complexity requirements
  - Skills are always at hand
  - Fast to execute
- So why not just use code?
  - Choice of language (/platform) variable
  - Code depends on external facilities – so not portable
  - Code is compiled, subject to deployment restrictions
  - Avoid developer dependency (more $$$)

Table Mapping forms

- Some form of column based table (spreadsheet, database)
- A very obvious thing to do
- Handles the 90% well (complements the code approach)
- Maps well between elements when
  - names are different
  - Value domains are very similar or assumed to be so (context dependent)
- Rapidly gets complicated for the other 10%
  - Frequently encounter sophisticated implied language in the columns

HL7 v2					HL7 FHIR				
Display Seq	Identifier	Name	Data T	Cardi	FHIR Attribute	Data Type	Cardinal	Data Type Mappi	Vocabulary Mapping	Comments
01.00.00	MSH-1	Field Separator	ST		N/A					Field separators are not used in FHIR. Either XML or
02.00.00	MSH-2	Encoding Characters	ST		N/A					Encoding characters are not used in FHIR.
03.00.00	MSH-3	Sending Application	HD	0..1	MessageHeader.source	BackboneElement	1..1	HD[MSH-3]		Example: MSH|^~\%|LAB^2.16.840.1.113883.3.987.1
04.00.00	MSH-4	Sending Facility	HD	0..1	MessageHeader.sender	Reference(Organiza	0..1	HD[Organization]		How do we populate this if MSH-4 is not valued?How c
05.00.00	MSH-5	Receiving Application	HD	0..1	MessageHeader.destination[first]	BackboneElement	0..*	HD[MSH-5]		Example: MSH...|RH^2.16.840.1.113883.632.3^ISO|.
06.00.00	MSH-6	Receiving Facility	HD	0..1	MessageHeader.destination[first].receiver	Reference(Organiza	0..1	HD[MSH-6]		
07.00.00	MSH-7	Date/Time of Message	DTM	1..1	MessageHeader.#ext-timestamp#	dateTime				If a message is always conveyed in a Bundle, then Bur
08.00.00	MSH-8	Security	ST	0..1	MessageHeader.meta.security	string	0..*			
09.00.00	MSH-9	Message Type	MSG	1..1	MessageHeader.event.coding	coding	1..1	MSG		Set up time with InM. Set up mapping tables for all com
10.00.00	MSH-10	Message Control ID	ST	1..1	Bundle.identifier	identifier	0..*	ST[Identifier]		Need a gForge to request a MessageHeader.identifier
11.00.00	MSH-11	Processing ID	PT	1..1	MessageHeader.meta.tag	coding	0..*	PT		
12.00.00	MSH-12	Version ID	VID	1..1	N/A					We agreed that the version of the v2 message that wa
13.00.00	MSH-13	Sequence Number	NM	0..1						
14.00.00	MSH-14	Continuation Pointer	ST	0..1	N/A					
15.00.00	MSH-15	Accept Acknowledgment Type	ID	0..1	MessageHeader.#ext-acceptAcknowledgementType#				Acknowledgements	Check what LRI/LOI terminology is and use that.We sh
16.00.00	MSH-16	Application Acknowledgment Type	ID	0..1	MessagaeHeader.#ext-applicationAcknowledgementType#				Acknowledgements	Check what LRI/LOI terminology is and use that.
17.00.00	MSH-17	Country Code	ID	0..1	MessageHeader.sender(Organization.address.country)				Countries	
18.00.00	MSH-18	Character Set	ID	0..*						
19.00.00	MSH-19	Principal Language Of Message	CWE	0..1	MessageHeader.language	code	0..1		Language	This does only cover the MessageHeader resource. N
20.00.00	MSH-20	Alternate Character Set Handling Scheme	ID	0..1						
21.00.00	MSH-21	Message Profile Identifier	EI	0..*	MessageHeader.definition(MessageDefinition)???			EI[MSH-21]		Need to support multiple profiles. Also, comment on im
22.00.00	MSH-22	Sending Responsible Organization	XON	0..1	MessageHeader.responsible	Reference(Organiza tion)	0..1	XON		
23.00.00	MSH-23	Receiving Responsible Organization	XON	0..1	MessageHeader.destination.receiver[first]	Reference(Organiza tion)	0..*	XON		This is an alternate representation of MSH-6. Need to
24.00.00	MSH-24	Sending Network Address	HD	0..1	Messageheader.source.endpoint	uri	0..1	HD[MSH-24]		
25.00.00	MSH-25	Receiving Network Address	HD	0..1	MessageHeader.destination[first].endpoint	uri	0..*	HD[MSH-25]		This is an alternate representation of MSH-5. Need to

https://docs.google.com/spreadsheets/d/13pgda5xl-PwCgB9j0axyymwwv7RJVcrIzY8Ah1y1Y1M/edit#gid=0

HL7 v2			Condition (IF True)	HL7 FHIR						
Name	Data Type	Cardin		FHIR Attribute	Data Type	Cardin	Data Type Mapping	Vocabulary Mappin	Derived Ma	Comments
Set ID - PID	SI	0..1		N/A						
Patient ID	CX	0..1		Patient.identifier	Identifier	0..*	CX			
Patient Identifier List	CX	1..*		Patient.identifier	Identifier	0..*	CX			
Alternate Patient ID - PID	CX	0..1		Patient.identifier	Identifier	0..*	CX			
Patient Name	XPN	1..*		Patient.name	HumanName	0..*	XPN			
Mother's Maiden Name	XPN	0..*		Patient[second].name	HumanName	0..*	XPN		Patient.na	This may not be the second name if there are re
Date/Time of Birth	DTM	0..1		Patient.birthdate	date	0..1				
Administrative Sex	CWE	0..1		Patient.gender	code	0..1	CWE[code]	Gender		
Patient Alias	XPN	0..1		Patient[third].name	HumanName	0..*	XPN			
Race	CWE	0..*		Patient.#ext-race#	coding	0..*	CWE[Coding]			For an international document, it may not be pos
Patient Address	XAD	0..*		Patient.address	Address	0..*	XAD			
County Code	IS	0..1		Patient.address[second].district	string	0..1				This may not be the second occurrence if PID-1
Phone Number - Home	XTN	0..*		Patient.telecom	ContactPoint	0..*	XTN			
Phone Number - Business	XTN	0..*		Patient.telecom	ContactPoint	0..*	XTN			
Primary Language	CWE	0..1		Patient.communication.language	CodeableCon	0..*	CWE[CodeableConcept]	Language		
Marital Status	CWE	0..1		Patient.maritalStatus	CodeableCon	0..1	CWE[CodeableConcept]	Martial Status		
Religion	CWE	0..1		Patient.ext-patient-religion	CodeableCon	0..*	CWE[CodeableConcept]	Religion		
Patient Account Number	CX	0..1		Account.identifier	Identifier	0..*	CX			Where Account.subject.Reference(Patient.ident
SSN Number - Patient	ST	0..1		Patient.identifier.value	string	0..1			Patient.ide	the value set for Identifier.type doesn't contain S
Driver's License Number - Patient		0..1		Patient.identifier.value	string	0..1	DLN			
Mother's Identifier	CX	0..*		RelatedPerson.identifier	Identifier	0..*	CX			Where RelatedPerson.patient.Reference(Patien
Ethnic Group	CWE	0..*		Patient.#ext-ethnicity#	coding	0..*	CWE[Coding]			For an international document, it may not be pos
Birth Place	ST	0..1		Patient.ext-birthplace.text	string	0..1				
Multiple Birth Indicator	ID	0..1	IF PID-25 DOES NOT EXIST	Patient.multipleBirthBoolean	boolean	0..1		Yes/No to boolean		
Birth Order	NM	0..1		Patient.multipleBirthInteger	integer	0..1				
Citizenship	CWE	0..*		Patient.ext-citizenship.code	CodeableCon	0..*	CWE[CodeableConcept]			The standard extension does not include a value
Veterans Military Status	CWE	0..1		Patient.#ext-veteranMilitaryStatus#	CodeableCon	0..1	CWE[CodeableConcept]			
Nationality	CWE	0..1		Patient.ext-nationality.code	CodeableCon	0..*	CWE[CodeableConcept]			The standard extension does not include a value
Patient Death Date and Time	DTM	0..1		Patient.deceasedDateTime	dateTime	0..1				
Patient Death Indicator	ID	0..1	IF PID-29 DOES NOT EXIST	Patient.deceasedBoolean	boolean	0..1		Yes/No to boolean		
Identity Unknown Indicator	ID	0..1		Patient.#ext-identityUnknown#	boolean	0..1		Yes/No to boolean		
Identity Reliability Code	CWE	0..*								
Last Update Date/Time	DTM	0..1		Patient.meta.lastUpdated	instant	0..1				
Last Update Facility	HD	0..1		Meta.#ext-lastUpdatedFacility#	Reference(Or	0..1				
Taxonomic Classification Code	CWE	0..1		Patient.ext-animal.species	CodeableCon	0..1	CWE[CodeableConcept]			A voacabulary map may not be worth the lift give
Breed Code	CWE	0..1		Patient.ext-animal.breed	CodeableCon	0..1	CWE[CodeableConcept]			A voacabulary map may not be worth the lift give
Strain	ST	0..1								If this is necessary, it should be an update to the
Production Class Code	CWE	0..2								
Tribal Citizenship	CWE	0..*		Patient.ext-citizenship.code	CodeableCon	0..*	CWE[CodeableConcept]			The standard extension does not include a value

Table Mappings: Limitations

- Table mappings are a very convenient way to approach the problem
  - Work very well for the 90% easy elements
  - Implicit grammar gets out of control for the next 9%
  - Can't address the last 1%
- Can't automate the transform based on this information
- So this is documentation to support code
  - (or maybe called from code)

- Corresponds (roughly) to ConceptMap – element mapping *in concept*

Concept Map

source		0..1	uri	Source system where concepts to be mapped are defined
sourceVersion		0..1	string	Specific version of the code system
target		0..1	uri	Target system that the concepts are to be mapped to
targetVersion		0..1	string	Specific version of the code system
element		1..*	BackboneElement	Mappings for a concept from the source set
code		0..1	code	Identifies element being mapped
display		0..1	string	Display for the code
target	I	0..*	BackboneElement	Concept in target system for element *+ Rule: If the map is narrower or inexact, there SHALL be some comments*
code		0..1	code	Code that identifies the target element
display		0..1	string	Display for the code
equivalence	?!	1..1	code	relatedto | equivalent | equal | wider | subsumes | narrower | specializes | inexact | unmatc ConceptMapEquivalence (Required)
comment	I	0..1	string	Description of status/issues in mapping
dependsOn		0..*	BackboneElement	Other elements required for this mapping (from context)
property		1..1	uri	Reference to property mapping depends on
system		0..1	canonical(CodeSystem)	Code System (if necessary)
value		1..1	string	Value of the referenced element
display		0..1	string	Display for the code (if value is a code)
product		0..*	see dependsOn	Other concepts that this mapping also produces

Table Mapping Approach

- Need to reflect this formally as a FHIR resource
  - Goal: Make mapping a community activity
- Use ConceptMap for now – needs work

FHIR Mapping Language

- Principles

Design Features

- Describe transform from one instance to another where instances are trees of elements
- Describe one way transforms
- Statements of relationship
  - no procedural features - supports meta analysis
- Structure/hierarchy based
  - e.g. support graphical transform builders
- Modular to allow re-use
- Can use types where they are present, but does not depend on them

Resource StructureMap

- A resource like all others
- Has the same metadata as other conformance resources
- Defined content is the abstract syntax tree for the mapping language
  - Technically, that's another concrete syntax
- Recommended narrative is the mapping language
- Tools can interconvert between the forms (FHIR Validator)
- http://test.fhir.org/r3+ supports text format

Source Structure

```
TLeft
  a : string [0..1]
```

Target Structure

```
TRight
  a : string [0..1]
```

```
map "http://hl7.org/fhir/StructureMap/tutorial" = tutorial

uses "http://hl7.org/fhir/StructureDefinition/tutorial-left" as source
uses "http://hl7.org/fhir/StructureDefinition/tutorial-right" as target

group tutorial
  input "source" : TLeft as source
  input "target" : TRight as target

// rules go here

endgroup
```

Fields with Different Names

Source Structure

```
TLeft
  a1 : string [0..1]
```

Target Structure

```
TRight
  a2 : string [0..1]
```

```
"rule_a1" : for source.a1 as b make target.a2 = b
```

Hierarchical Content #3


```
TLeft
  az1 :[0..1]
    az2 : string [1..1]
    az3 : string [0..*]
```

```
TRight
  az1 :[0..*]
    az2 : string [1..1]
    az3 : string [0..1]
```

```
// setting up a variable for the parent
aza : for src.az1 as s_az1 then {

  // one target.az1 for each az3
  azb : for s_az1.az3 as s_az3 make target.az1 as t_az1 then {
    // value for az2. Note that this refers to a previous context in the source
    az2 : for s_az1.az2 as az2 make t_az1.az2 = az2

    // value for az3
    az3 : for s_az3 make tgt_az1.az3 = src_az3
  }
}
```

Examples

- FHIR Version Transforms:
  - http://hl7.org/fhir/r3maps.html
  - https://github.com/FHIR/interversion - ongoing

- Prototype CDA Maps:
  - https://build.fhir.org/ig/HL7/fhir-cda
  - https://github.com/jduteau/ccda-to-fhir-maps

Mapping Language

- Very Portable
- Total ban on Procedural Code is conceptually challenging
  - Allows for Meta-analysis
- Adoption is very slow – but real
- Key feature: the API

Mapping Language API

- ValueSet validation operation
- Translation operation
- Lookup another tree of data
- Create an instance tree
- Return the correct string format to refer to a tree (input or output)

- This is what creates the portability

Javascript Sandbox

- Proposal – Documentation:
  - https://github.com/FHIR/interversion/blob/master/engine/readme.md

- Standard sandbox that can be implemented by any application. E.g.
  - Integrated engine inside interface engine
  - Canned Lambda function
  - In any clinical application

- Requires a standard Javascript engine

Sandbox Entry Point

```
function convert(services, object, api) {
}
```

Parameters:

- services: an object that makes conversion/transformation services available to the script - see below for documentation
- object: the source object being converted
- api: a FHIR Client - provides direct access to the FHIR persistence store, pre-authorized

Sandbox Services - Terminology

```
function lookup(coded, params) : Parameters;
function translate(conceptMap, code, params) : Parameters;
function expand(valueSet, params) : ValueSet;
function validateVS(valueSet, coded, params) : Parameters;
function validateCS(codeSystem, coded, params) : Parameters;
function subsumes(system, coded1, coded2) : code;
function translateCode(code, srcSystem, dstSystem) : String;
```

Sandbox Services - Conversion

```
function factory(typeName) : Object;
function runJS(scriptName, routineName, params....) : Object | void;
function runMap(url, source[, target], callBacks...)
function runLiquid(fileName, source, type[, format]) : Object;
function runMarkdown(fileName, source) : Object;
function translateUri((value, type)) : String;
function translateDate(date, srcFmt, dstFmt) : String;
```

Other – MDMI? QVT? XSLT?

Example

```
// for use with ADT_A01 message
function convert(services, object, api) {
  // first step: process the patient
  var pid = object.segment[2];
  var patid = pid.field[3].element[1].text;
  // or it could be:  patid = pid.q('field[3].element.where(component[5] = "MR")').text;
  var pat = api.read('Patient', patid, patid); // assuming that we store patients with MYN as master
  if (pat == null)
    pat = makePatient(services, pid, api);
  else
    updatePatient(pat, pid, api);
  // now: process the encounter
}
```

Example

```
function makePatient(services, pid, api, patid) {
  // use a liquid script to make the patient resource
  var pat = services.liquid("pid.liquid", pid, "Patient", "json");

  // doing this in the code here rather than the liquid script is a design choice;
  // the id might not always be the same, or setting it in the liquid template might
  //make the liquid template less reusable
  pat.id = patid;

  return api.update(pat);
}
```

Methods for Transforming content

- Work in progress
- None of the options are perfect (yet)

- Need to focus on making it a community approach
  - Not a solitary obsession