

Mapping V2 to FHIR

Simone Heckmann

Chief Technical Officer

sh@gefyra.de

0177 39 39 36 7

www.gefyra.de



Who we are



„gefyra“ is the Greek word for „bridge“

We offer

- FHIR training,
- consultancy for FHIR projects
- professional tooling
- integration services (j/w Health-Comm, based on INfOR Cloverleaf®)

We have been involved with FHIR since 2014,
have tested our V2 mappings on > 7 Connectathons,
and have them running in production environments



Scenario

Mapping a V2 ADT_A01 message

http://wiki.hl7.org/index.php?title=Version_2_-_FHIR_Mapping_Scenarios


















MSH|^~\&|LegacyEHR||SPARK||20150502090000||ADT^A01|00000002|P|2.5
EVN|A01|20150502090000|
PID|1||345345412312345^10^^NHS^NH~456756756745^^^TCPAS^MR||Gefyra^Alpha^F^^Ms^
^L||19280524|F|||Bridge Street 22^^Bridgetown^^^12345^^^P||277543^PRN|||U||
NK1|1|Gefyra^Beta|FTH|||+44 201 12345678||
PV1||I|INT^0001^02^GENHOS|||0100^ANDERSON,CARL|0148^ADDISON,JAMES||SUR|||
0148^ANDERSON,CARL|S|234637^^^GENHOS|A|||GENHOS|||2015050209
0000|
AL1|1|DA|1605^acetaminophen^L|MO|Muscle Pain~hair loss
AL1|2|DA|1558^Oxycodone^L|MO|Muscle Pain~hair loss
AL1|3|MA|2221^Peanuts^L|SV|Anaphylactic Shock

MSH ^~\& LegacyEHR SPARK 20150502090000 ADT^A01 00000002 P 2.5	MessageHeader
EVN A01 20150502090000	
PID 1 345345412312345^10^^NHS^NH~456756756745^^^TCPAS^MR Gefyra^Alpha^F^^Ms^	Patient
^L 19280524 F Bridge Street 224^Bridge Town^^12345^^^P 277543^PRN U	
NK1 1 Gefyra^Beta FTH +44 201 12345678	Encounter
PV1 I INT^0001^02^GENHOS 0100^ANDERSON,CARL 0148^ADDISON,JAMES SUR	
0148^ANDERSON,CARL S 234637^^^GENHOS A GENHOS 20150502090000	
AL1 1 DA 1605^acetaminophen^L MO Muscle Pain~hair loss	AllergyIntolerance(s)
AL1 2 DA 1558^Oxycodone^L MO Muscle Pain~hair loss	
AL1 3 MA 2221^Peanuts^L SV Anaphylactic Shock	

...so, it's a Bundle of Resources!

Yes. But there are multiple types of Bundles.
Is it a message, a batch, a transaction or a document...?

<http://build.fhir.org/bundle.html>

Name	Flags	Card.	Type	Description & Constraints
 Bundle	Σ I		Resource	Contains a collection of resources <i>FullUrl must be unique in a bundle, or else entries with the same fullUrl must have different meta.versionId</i> <i>entry.request only for some types of bundles</i> <i>entry.response only for some types of bundles</i> <i>total only when a search or history</i> <i>entry.search only when a search</i>
...  type	Σ	1..1	code	document message transaction transaction-response batch batch-response history searchset collection BundleType (Required)
...  total	Σ I	0..1	unsignedInt	If search, the total number of matches
...  link	Σ	0..*	BackboneElement	Links related to this Bundle
...  relation	Σ	1..1	string	http://www.iana.org/assignments/link-relations/link-relations.xhtml 
...  url	Σ	1..1	uri	Reference details for the link
...  entry	Σ I	0..*	BackboneElement	Entry in the bundle - will have a resource, or information <i>fullUrl cannot be a version specific reference</i> <i>must be a resource unless there's a request or response</i> <i>The fullUrl element must be present when a resource is present, and not present otherwise</i>
...  link	Σ	0..*	see link	Links related to this entry
...  fullUrl	Σ	0..1	uri	Absolute URL for resource (server address, or UUID/OID)
...  resource	Σ	0..1	Resource	A resource in the bundle
...  search	Σ I	0..1	BackboneElement	Search related information
...  mode	Σ	0..1	code	match include outcome - why this is in the result set SearchEntryMode (Required)
...  score	Σ	0..1	decimal	Search ranking (between 0 and 1)
...  request	Σ I	0..1	BackboneElement	Transaction Related Information
...  method	Σ	1..1	code	GET POST PUT DELETE HTTPVerb (Required)
...  url	Σ	1..1	uri	URL for HTTP equivalent of this entry

A message, obviously!















True, but which event? Which structure? What's the logic associated with the event? Does the server know all this?

Do we really want to re-implement V2 messaging with FHIR?

So what do we do?

We (the integration engine) know what the events mean and which actions we expect the server to perform. So we can *tell* the server what to do.

Let's try Transactions!

 entry	Σ I	0..*	BackboneElement	Entry in the bundle - will have a resource, or information <i>must be a resource unless there's a transaction or transaction response</i> <i>The fullUrl element must be present when a resource is present, and not present otherwise</i>
 link	Σ	0..*	see link	Links related to this entry
 fullUrl	Σ	0..1	uri	Absolute URL for resource (server address, or UUID/OID)
 resource	Σ	0..1	Resource	A resource in the bundle
 search	Σ I	0..1	BackboneElement	Search related information
 mode	Σ	0..1	code	match include outcome - why this is in the result set SearchEntryMode (Required)
 score	Σ	0..1	decimal	Search ranking (between 0 and 1)
 request	Σ I	0..1	BackboneElement	Transaction Related Information
 method	Σ	1..1	code	GET POST PUT DELETE HTTPVerb (Required)
 url	Σ	1..1	uri	URL for HTTP equivalent of this entry
 ifNoneMatch	Σ	0..1	string	For managing cache currency
 ifModifiedSince	Σ	0..1	instant	For managing update contention
 ifMatch	Σ	0..1	string	For managing update contention
 ifNoneExist	Σ	0..1	string	For conditional creates

what

how

```
MSH|^~\&|LegacyEHR||SPARK||20150502090000||ADT^A01|00000002|P|2.5
EVN|A01|20150502090000|
```

Patient -> create/update?

Encounter -> create

ALI	I	DA	1605^acetaminophen^L	MO	Muscle Pain~hair loss
-----	---	----	----------------------	----	-----------------------

AllergyIntolerance(s) -> purge/create



Now, let's REST-ify that!

How do I update/create?

We need to check if the Patient already exists. If it does, we want to PUT, if it doesn't we want to POST.

Enter the „Conditional UPDATE“!

<http://build.fhir.org/http.html#2.42.0.10.2>

2.42.0.10.2 Conditional update

The conditional update interaction allows a client to update an existing resource based on some identification criteria, rather than by [logical id](#). To accomplish this, the client issues a `PUT` as shown:

```
PUT [base]/[type]?[search parameters]
```

When the server processes this update, it performs a search using its standard [search facilities](#) for the resource type, with the goal of resolving a single logical id for this request. The action it takes depends on how many matches are found:

- **No matches:** The server performs a `create` interaction
- **One Match:** The server performs the update against the matching resource
- **Multiple matches:** The server returns a `412` Precondition Failed error indicating the client's criteria were not selective enough

This variant can be used to allow a stateless client (such as an interface engine) to submit updated results to a server, without having to remember the logical ids that the server has assigned. For example, a client updating the status of a lab result from "preliminary" to "final" might submit the finalized result using `PUT path/Observation?identifier=http://my-lab-system|123`

So what's the criteria?

The Conditional UPDATE will fail if it matches multiple resources, so our criteria must identify the Patient uniquely.

Sounds like „identifier“, right?

<http://build.fhir.org/search.html#token>

2.42.1.4.10 token

A token type is a parameter that searches on a URI/value pair. It is used against a code or identifier data type where the value may have a URI that scopes its meaning. The search is performed against the pair from a Coding or an Identifier. Matches are literal (e.g. not based on subsumption or other code system features), but not case sensitive. To use subsumption based logic, use the modifiers below, or list all the codes in the hierarchy. The syntax for the value is one of the following:

- **[parameter]=[code]** : the value of **[code]** matches a Coding.code or Identifier.value irrespective of the value of the system property
- **[parameter]=[system]|[code]** : the value of **[code]** matches a Coding.code or Identifier.value, and the value of **[system]** matches the system property of the Identifier or Coding
- **[parameter]=|[code]** : the value of **[code]** matches a Coding.code or Identifier.value, and the Coding/Identifier has no system property
- **[parameter]=[system]|** : any element where the value of **[system]** matches the system property of the Identifier or Coding

Note: The namespace URI and code both must be **escaped** correctly.

PID|1||**345345412312345**^10^^**NHS**^NH~456756756745^^^TCPAS^MR||



urn:oid:2.16.840.1.113883.2.1.4.1



<http://www.ghh.org/identifiers>

How do I purge/create?

First, we need to DELETE all previously submitted AllergyIntolerances, then POST the new ones. But wait! We need to make sure, we only delete the ones we (the integration engine) created!

Enter the „Conditional Delete“!

<http://build.fhir.org/http.html#2.42.0.12.1>

So what's the criteria?

Somehow we need to be able to recognize the AllergyIntolerances we submitted.
This can be achieved by either

- adding tags to the resource metadata (simple) or
- using Provenance resources (not so simple, but more powerful)

<http://build.fhir.org/resource.html#Meta>

<http://build.fhir.org/provenance.html>

2.42.0.12.1 Conditional delete

The conditional delete interaction allows a client to delete an existing resource based on some selection criteria, rather than by a specific [logical id](#). To accomplish this, the client issues an HTTP `DELETE` as shown:

```
DELETE [base]/[type]/?[search parameters]
```

When the server processes this delete, it performs a search as specified using the standard [search facilities](#) for the resource type. The action it takes depends on how many matches are found:

- **No matches** or **One Match**: The server performs an ordinary `delete` on the matching resource
- **Multiple matches**: Servers may choose to delete all the matching resources, or it may choose to return a `412` Precondition Failed error indicating the client's criteria were not selective enough. A server indicates whether it can delete multiple resources in its [Capability Statement](#) (`.rest.resource.conditionalDelete`). if there are multiple matches, either all must be deleted, or the server SHALL return an error













This variant can be used to allow a stateless client (such as an interface engine) to delete a resource on a server, without having to remember the logical ids that the server has assigned. For example, a client deleting a lab atomic result might delete the resource using `DELETE /Observation?identifier=http://my-lab-system|123`.

So, if we use tags, our criteria is:

GET [base]/AllergyIntolerance?patient.identifier=[?]&_tag=[?]

„chained search“

„token parameter“

Name	Flags	Card.	Type	Description & Constraints	
 Provenance	Σ		DomainResource	Who, What, When for a set of resources	
...  target	Σ	1..*			usually version specific)
...  period	Σ	0..1	Period	When the activity occurred	
...  recorded	Σ	1..1	instant	When the activity was recorded / updated	
...  reason	Σ	0..*	Coding	Reason the activity is occurring	
...  activity	Σ	0..1	Coding	Activity that occurred	
...  location	Σ	0..1	Reference(Location)	Where the activity occurred, if relevant	
...  policy	Σ	0..*	uri	Policy or plan the activity was defined by	
...  agent	Σ	1..*			
...  role	Σ	1..1	Coding	what the agents involvement was	
...  actor	Σ	0..1	Reference(Practitioner RelatedPerson Patient Device Organization)	Individual, device or organization playing role	
...  userId	Σ	0..1	Identifier	Authorization-system identifier for the agent	

2.42.1.4.14 Reverse Chaining

The `_has` parameter provides limited support for reverse chaining - that is, selecting resources based on the properties of resources that refer to them (instead of chaining, above, where resources can be selected based on the properties of resources that they refer to). Here is an example of the `_has` parameter:

```
GET [base]/Patient?_has:Observation:patient:code=1234-5
```

This requests the server to return Patient resources, where the patient resource is referred to by at least one Observation where the observation has a code of 1234, and where the Observation refers to the patient resource in the patient search parameter.

Note the following limitations on the use of the `_has` parameter:

- There is no support for reverse chaining to more than one level
- Modifiers are not supported (e.g. GET [base]/Patient?_has:Observation:subject:code:in=http://loinc.org)
- Chaining is not supported (e.g. GET [base]/Patient?_has:Observation:subject:device.identifier=foo)

So, in our case:

```
GET [base]/AllergyIntolerance?patient.identifier=[?]&_has:Provenance:agent:device=[?]
```

We also need to create a **Device** resource!

Advantage: the **Provenance** Resource also gives us an chance to store the original message for traceability as an **Attachment**. (...yes, we need to create *that* resource, too...)

Does that *really* work?

Well, we *made* it work (at least, with tags)
after changing the processing order for Transactions to
DELETE > POST > PUT > GET

<http://build.fhir.org/http.html#2.42.0.16.2>

Now: the field mapping!

Check the „mapping“ tab at the top of each resource

There's no „one-size-fits-all“ – V2 structures and FHIR server requirements will vary!

+ look at the FHIR mapping language and StructureMap resource!

+ look at ConceptMap resource and the \$translate-Operation!

<http://build.fhir.org/mapping-language.html>

<http://build.fhir.org/structuremap.html>

<http://build.fhir.org/conceptmap.html>

<http://build.fhir.org/terminology-service.html#4.6.8>

How do I reference a resource that doesn't have a url (yet)?

Assign a UUID to every Bundle.entry.fullUrl

Then reference the associated resource by this uri.

The server must replace UUIDs with the actual urls when processing the Transaction

<http://build.fhir.org/bundle.html#references>


```
<!-- A patient that doesn't have a persistent home - but it does have  
a UUID assigned for this bundle "locally identified" -->
```

```
<entry>
```

```
  <fullUrl value="urn:uuid:04121321-4af5-424c-a0e1-ed3aab1c349d"/>
```

```
  <resource>
```

```
    <Patient>
```

```
    </Patient>
```

```
  </resource>
```

```
</entry>
```

```
<!-- reference to a locally identified resource -->
```

```
<entry>
```

```
  <fullUrl value="http://example.org/fhir/Observation/12"/>
```

```
  <resource>
```

```
    <Observation>
```

```
      <id value="12"/>
```

```
      <subject>
```

```
        <!-- reference to the second patient above -->
```

```
        <reference value="urn:uuid:04121321-4af5-424c-a0e1-ed3aab1c349d"/>
```

```
      </subject>
```

```
    </Observation>
```

```
  </resource>
```

```
</entry>
```



But what if I don't want to touch the resource I need to reference?

Conditional Update will update the matching resource, Conditional Create will create a new resource if none matches. Both can get us into trouble, if we don't have permission to create/update specific resources.

Solution: Use Conditional References instead!

<http://build.fhir.org/bundle.html#references>

Conditional References

When constructing the batch, the client may not know the logical id of a resource, but it may know identifying information - e.g. an identifier. This situations arises commonly when building transactions from v2 messages. The client could resolve that identifier to a logical id using a search, but that would mean that the resolution to a logical id does not occur within the same transaction as the commit (as well as significantly complicating the client). Because of this, in a transaction (and only in a trasaction), references to resources may be replaced by a search URI that describes how to find the correct reference:

```
<Bundle xmlns="http://hl7.org/fhir">
  <id value="20160113160203" />
  <type value="transaction" />
  <entry>
    <fullUrl value="urn:uuid:c72aa430-2ddc-456e-7a09-dea8264671d8" />
    <resource>
      <Observation>
        <subject>
          <reference value="Patient?identifier=12345"/>
        </subject>
        <!--! rest of resource omitted -->
      </Observation>
    </resource>
    <request>
      <method value="POST" />
    </request>
  </entry>
</Bundle>
```

Limitations!

- V2 Source will overwrite all changes and additions to Resources on the Server (-> Patch)
- There is no „trigger“ to invoke additional actions on the server (-> Message/Operations)
- „Merge“ is tricky (-> Operations)
- Doesn't work in the absence of REST Protocol (-> Message)
- Is it PV1 actually an Encounter or rather an EpisodeOfCare?

What if the server doesn't support Transactions?

Basically, the Transaction Bundle constitutes a list of individual, atomic REST interactions. We can run them through „post processing“ and split them up.

But remember: we have to handle the replacing of the UUIDs with the actual URLs ourselves!

We can even split the Conditional interactions up into a GET and PUT/POST/DELETE interaction,

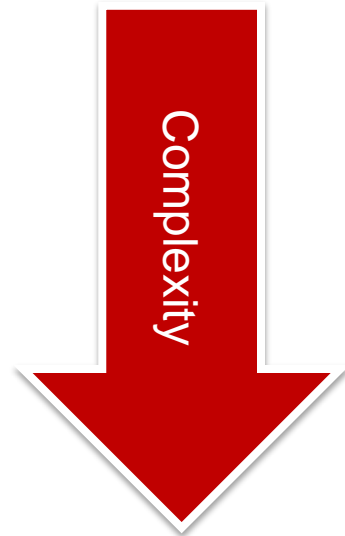
But then we have to deal with the errors on our side.

You can move complexity around, but
you can't make it go away

Grahame Grieve

Client

Server



Message

Transaction

Plain REST



What next?

- Tell us about your use case and join the Hands On
- Give feedback
- Join the discussion on how to handle *merge*
- Join the next connectathon in San Antonio and help us to evaluate Subscription and _history interactions to transfer Resources from a FHIR server back to a V2 system

http://wiki.hl7.org/index.php?title=201701_Resource_Subscription_Track



gefyrda

we make fhir[®] work