Adding IRI Stems to the HL7 Terminology Website to Support Concept IRIs

A proposal from the RDF Subgroup of the HL7 Implementable Technology Specifications (ITS) group, a joint collaboration between HL7 and W3C

22-Jun-2022 (modified Jan-2023)
https://bit.ly/fhir-rdf-concept-iris-july-2022

Abstract

Concepts in a FHIR terminology are currently identified by a pair of identifiers: `Coding.system` and `Coding.code`, such as `http://snomed.info/sct` and `128045006`. However, the Web Architecture recommends — and many applications prefer — to use a single URI as a unique identifier for each concept. We call these *Concept URIs*, or more generally, *Concept IRIs* if they are Internationalized Resource Identifiers (IRIs) instead of URIs. Concept IRIs are formed by combining an *IRI Stem* with a `Coding.code`, typically by simple concatenation. For example, IRI Stem `http://snomed.info/id/` is concatenated with `128045006` to form the Concept IRI `http://snomed.info/id/128045006`. A terminology's IRI Stem is uniquely related to its `Coding.system`, but the two are not necessarily the same, and there is no standard syntactic correspondence between them: a mapping is required. For interoperability, it would be helpful if HL7 provided a standard place to register and lookup these mappings. **We propose** that mappings between `Coding.systems` and IRI Stems should be stored in the HL7 Terminology website. This would allow both humans and software tools to conveniently find these mappings to convert FHIR Codings to or from Concept IRIs. The RDF subgroup of the HL7 Implementable Technology Standards (ITS) working group successfully implemented and tested this approach on a fork of the GitHub repository that generates the HL7 Terminology website.

Background: Concept IRIs and IRI Stems

FHIR's resource-oriented architecture is based on URIs and URLs for everything except Codings and Quantities (e.g. Resources, Datatypes, Valuesets, Profiles, etc.). FHIR terminology concepts are identified by `Coding.system-Coding.code` pairs. To support applications that prefer to use a single URI — or more generally a single Internationalized Resource Identifier (IRI) — to uniquely identify each concept, a **Concept IRI** corresponding to each `Coding.system-Coding.code` pair can be used instead. The Concept IRI for a given `Coding.system-Coding.code` pair is formed by combining the ***IRI Stem*** for that terminology with the `Coding.code`, typically by simple concatenation, though percent-encoding may be required for special characters (explained below). It would be convenient if the IRI Stem were the same as the `Coding.system`, but unfortunately there is no standard syntactic relationship between them. (Table 1 below shows examples.) Consequently, we need a standard place to store mappings between `Coding.systems` and IRI Stems, and the HL7 Terminology website is the logical place.

`Coding.system`	**`Coding.code`**	**IRI Stem**	**Concept IRI**
ICD 10: `http://hl7.org/fhir/sid/icd-10`	`G44.1`	`http://purl.bioontology.org/ontology/ICD10/`	`http://purl.bioontology.org/ontology/ICD10/G44.1`
SNOMED CT: `http://snomed.info/sct`	`128045006`	`http://snomed.info/id/`	`http://snomed.info/id/128045006`
SNOMED CT: `http://snomed.info/sct`	`128045006:{363698007=56459004}`	`http://snomed.info/id/`	`http://snomed.info/id/128045006:{363698007=56459004}`
MeSH: `https://www.nlm.nih.gov/mesh`	`D000305`	`http://id.nlm.nih.gov/mesh/`	`http://id.nlm.nih.gov/mesh/D000305`
LOINC: `http://loinc.org`	`35217-9`	`https://loinc.org/rdf/`	`https://loinc.org/rdf/35217-9`
Example coding system that uses a Unicode smiling face character (U+263A) as a code: `http://example.org/`	☺	`http://example.org/`	`http://example.org/☺`
Example coding system that uses a Unicode waving hand character (U+1F44B) from the *Miscellaneous Symbols and Pictographs block*, combined with the medium-dark skin tone (U+1F3FE): `http://example.org/`	👋🏾	`http://example.org/`	`http://example.org/id/👋🏾`

Table 1. Some example FHIR `Coding.systems`, corresponding IRI Stems and Concept IRIs.

Storing IRI Stems on the HL7 Terminology website

We propose that mappings between `Coding.systems` and IRI Stems be stored in the *CodeSystem* and *NamingSystem* records in the HL7 Terminology website and made available in the corresponding NPM package. This will enable any software tool to conveniently find these mappings to convert FHIR Codings to or from Concept IRIs.

The HL7 Terminology website is the authoritative source for HL7 CodeSystems and related NamingSystems. It is maintained by the FHIR Unified Terminology Governance (UTG) Project and generated from the UTG GitHub repository. The HL7 Terminology website makes CodeSystem and NamingSystem resources available in machine-readable formats like XML, JSON and Turtle. The XML format is authoritative; other formats are generated from the XML by the _genonce.sh script included in the same repository. The data can also be accessed programmatically via the Node Package Manager (NPM), as HL7 Terminology publishes a *hl7-terminology* NPM package that includes JSON representations of CodeSystems (see example) and NamingSystems (see example). All changes to the HL7/UTG repository are tracked using the Git version control system. It is therefore the perfect place to include IRI Stem information.

To test this, we created a draft pull request to the UTG repository in which we added IRI Stem information to some CodeSystems (see SNOMED-CT as an example) and NamingSystems (see SNOMED-CT as an example). We wrote a small library for accessing these IRI Stem values from within the hl7-terminology NPM package, and wrote a small test script that demonstrates how this library can be used to convert FHIR system/code pairs to Concept IRIs and vice versa, including converting some examples included with the FHIR examples. This worked successfully. We then created a UTG proposal (UP-364) in which we made the followed changes:

1. For *CodeSystem*s, we set the *Identifier.system* to `urn:ietf:rfc:3987`, *Identifier.type* to a Coding with *system*=`http://terminology.hl7.org/CodeSystem/v2-0203` and *code*=`IRISTEM` (as proposed in UP-364), and the *Identifier.value* to the IRI Stem (example).
2. For *NamingSystem*, we could not set the *uniqueId.type* to `iri-stem` as the values in this field must come from the *NamingSystemIdentifierType* enumeration. Instead, we set *uniqueId.comment* to `IRIstem` and *uniqueId.value* to the IRI Stem (example).

We propose that these XML representations be supplemented with IRI Stems:

1. By designating "`urn:ietf:rfc:3987`" as an *Identifier.system* value for IRIs in https://build.fhir.org/identifier-registry.html, as proposed in FHIR-37960.
2. By adding an "`iri-stem`" value to the *NamingSystemIdentifierType*, allowing it to be used as a value in the *NamingSystem.uniqueId.type* field in NamingSystems, as proposed in FHIR-39604.

Algorithm for creating a Concept IRI

This section defines a standard algorithm for generating a Concept IRI from a `<Coding.system, Coding.code>` pair. In many cases it involves merely concatenating the associated IRI Stem with the `Coding.code`. But because a `Coding.code` could contain reserved characters that are used to delineate different parts of the IRI, percent-encoding of reserved characters is required, as defined below.

Given:

- a FHIR `Coding.system`*, s,* that identifies a terminology *t*; and
- a `Coding.code`, *c*, that is defined within *t*;

a Concept IRI, *conceptIRI*, corresponding to *s* and *c* is computed as follows:

1. If no IRI Stem is defined for *s* in the HL7 Terminology website, then *conceptIRI* is undefined. Halt.
2. Let *iStem* be an IRI Stem that is defined for *s* in the HL7 Terminology website.
3. As a special case, if *iStem* equals `urn:ietf:rfc:3987`, then *conceptIRI* is *c,* and *c* MUST be a syntactically valid absolute-IRI as defined by RFC 3987. Halt. *(Non-normative comments: The purpose of this special case is to permit System.codes that are already IRIs to be used directly as Concept IRIs, without any transformation. Note that an absolute-IRI may also be a URL or a URN.)*
4. Let *cSafe* be the IRI-safe version of *c*, as defined by the algorithm in section 7.3 of R2RML: RDB to RDF Mapping Language (W3C Recommendation 27 September 2012), non-normatively quoted here for convenience:

   > *"The **IRI-safe version** of a string is obtained by applying the following transformation to any character that is not in the `iunreserved` production in [RFC3987]:*
   >
   > > *1. Convert the character to a sequence of one or more octets using UTF-8 [RFC3629]*
   > > *2. Percent-encode each octet [RFC3986]"*

   The `iunreserved` production defined in RFC 3987, section 2.2 using ABNF is also non-normatively quoted here for convenience:

   ```
   iunreserved    = ALPHA / DIGIT / "-" / "." / "_" / "~" / ucschar
   ```

   The `ucschar` production defined in RFC 3987, section 2.2 is also non-normatively quoted here for convenience. (*Non-normative comment: The ucschar production defines international character ranges that are valid unicode characters within the intersection of path components (ipath), query strings (iquery) and fragment identifiers (ifragment). They do not include any reserved characters involved in parsing apart the various components of an IRI.*)

```
        ucschar        = %xA0-D7FF / %xF900-FDCF / %xFDF0-FFEF
                          / %x10000-1FFFD / %x20000-2FFFD / %x30000-3FFFD
                          / %x40000-4FFFD / %x50000-5FFFD / %x60000-6FFFD
                          / %x70000-7FFFD / %x80000-8FFFD / %x90000-9FFFD
                          / %xA0000-AFFFD / %xB0000-BFFFD / %xC0000-CFFFD
                          / %xD0000-DFFFD / %xE1000-EFFFD
```

5. *conceptIRI* is the result of concatenating *iStem* and *cSafe*.

Some real and hypothetical examples are shown in Table 1 above.

Terminology versioning. In addition to the `Coding.system` and `Coding.code` fields, a Coding object includes three other fields: *display* (a human-readable label for the code), *userSelected* (a boolean field indicating whether this coding was chosen by the user), and *version* (the version of the system being referenced). In theory, one might wish to generate a separate Concept IRI for each `Coding.code` in each version of a terminology, which would mean encoding the version into the Concept IRI. However, experience with terminology versioning has shown that this is not usually helpful. In the algorithm above we have therefore opted to omit the version from the Concept IRI. If use cases requiring the version number in the Concept IRI emerge, the above algorithm could be later extended to construct Concept IRIs with version identifiers.

Security considerations

Since IRI Stems are based on IRIs, the same security considerations that apply to IRIs also apply to IRI Stems.

Another consideration is that a malicious actor may convince the UTG group to include an incorrect IRI Stem for a particular CodeSystem, potentially by compromising the HL7 Terminology website or the login account of a UTG group member. For example, the malicious actor might cause “`http://malicious.actor.org/`” to be stored in the HL7 Terminology website, as the IRI Stem for MeSH. Application software that looks up the IRI Stem for MeSH may then be tricked into forming malicious Concept IRIs, such as "`http://malicious.actor.org/D000328`", that point to a website controlled by the malicious actor. This could cause three problems:

1. If a healthcare application tries to dereference a malicious Concept IRI such as "`http://malicious.actor.org/D000328`" it may unknowingly download malicious data.
2. If a healthcare application dereferences resulting Concept IRIs, the malicious actor could surreptitiously monitor which MeSH concepts were being used by that application. This could allow the malicious actor to deduce which concepts were involved. For example, if the malicious actor knew when a particular patient was accessing his/her healthcare

application (perhaps by inducing the patient to do so, via forged email or text message), and the healthcare application dereferenced the malicious Concept IRIs, then by monitoring which malicious Concept IRIs had been dereferenced, the malicious actor could deduce healthcare information about that patient. Even if the malicious actor did not know which patient was accessing the healthcare application, the malicious actor could still monitor the frequency distribution of the concepts being dereferenced.

3. If some systems use the malicious IRI Stem and others use the correct IRI Stem, interoperability between those systems would be impaired, potentially resulting in denial of service.

Cost / Benefit Analysis

Benefits:

- **Standard Concept IRIs**: In accordance with Web Architecture, this proposal would enable any applications that use web-oriented Concept IRIs to have standard, globally unique IRIs for all concepts in all FHIR terminologies. This is important for any RDF applications that work with FHIR data. It also allows other applications the simplicity of using a single IRI to uniquely identify each concept.

Cost:

- **Initial implementation:** The RDF Subgroup has already tested the proposed solution, so it is already known to work. To implement it, the RDF Subgroup will submit pull requests to the UTG GitHub repository, both to extend any relevant FHIR documentation, and to initially populate the repository with IRI Stems for existing terminologies.
- **Maintenance:** IRI Stems tend to be very stable over time — much like `Coding.systems` — so very few changes are anticipated in the coming years, aside from adding more IRI Stems as new terminologies are defined. When changes are necessary, existing FHIR fields -- such as *Identifier.use* and *Identifier.period* -- could be used to differentiate between IRI Stems that are currently in use versus those that were in use previously.
- **Possibility of confusion:** There is a small risk that some users may confuse IRI Stems with `Coding.systems`. But given that `Coding.systems` are so heavily used in FHIR, such confusion seems unlikely.