Keith on STAMP

Here is some info on (Status, Time, Author, Module, Path) STAMP. We use a standard for managing fine-grained change to data artifacts, where we add new STAMPed versions of artifacts, resulting in a STAMP Chronology.

The chronology provides a means to generically represent the revisions to a component over time, and to index those revisions by status (active, inactive), effective time of change, author of change, module within which the change occurred (international edition, US extension, AMA extension, etc.), and the development path of the change (development, release candidate, etc.). Taken together, these fields can be referred to as a versions STAMP (status, time, author, module, and path).

STAMP is an amalgamation of widely used concepts that provides simplification and standardization. I’ve used it extensively, and the IHTSDO has as well in its tooling efforts.

See for example,  Table 88: Module Dependency Reference Set - Data Structure in the SNOMED technical implementation guide.

<http://ihtsdo.org/fileadmin/user_upload/doc/en_us/tig.html?t=trg2rfs_spec_module_depend>

This particular reference is important for a couple of reasons:

* Notice that objects are identified by UUIDs as the “identified object”. This is the same as I proposed: Identified objects using UUIDs as the primary identifier
* Notice the fields: active, effective time, module id. These correspond to the ST\*M\* of STAMP. They use the entire STAMP internally, but on release, they strip the author and path…
* Notice that this particular refset—the module dependency reference set—is the way SNOMED provides for representing dependencies between modules, taking account of module versioning. This is how we should initially approach “OSGi like” dependency declaration between modules.

Path has the meaning defined here for branch:

<https://en.wikipedia.org/wiki/Branching_(version_control)>

The problem with branch, is that it is technically incorrect for our use (and does not fit the STAMP acronym). A branch can have only one origin, where a path can have multiple origins. We frequently use multiple origins to represent a development path that incorporates more that one module (for example a version of SNOMED, and a version of the US extension, and a version of LOINC… 3 origins to a development path).

For more background, see for example Feiler’s paper on “Configuration management in Commercial Environments”

<http://pagesperso.lina.univ-nantes.fr/~molli-p/pmwiki/uploads/Main/feiler91.pdf>

A few snippets:

A version branch is used for several purposes:

· To represent an independent **path** of development, i.e., the maintenance of a

component as part of a field release vs. its further development

The result of a series of sequential changes is a

sequence of configuration versions, referred to as development **path**.

There are many more references to path in this document. This document is also the motivation for the change-set configuration management model applied to terminology, so it hold a special place in the history of SNOMED development.