# 2013.04.23 HL7 Templates Workgroup Minutes

## Attendees: Jane Curry, Kai Heitmann, Lisa Nelson, John Roberts, Mark Shafarman, Andy Stecheshin

## Scribe: Mark Shafarman

## Agenda Topics and Notes:

### Updates to May WG Schedule:

### for Wednesday , Q4: we need to add the joint meeting with FHIR, Vocab, etc. (Done)

### Jane will not be attending the WGM but we will try to include Jane at that meeting via Skype.

### John and Lisa will not be present at the Monday Q1 meeting, but will coordinate on issues via email before then as needed;

### Coordination items:

### HingX: Jane will coordinate with Andy to produce a status update for the WGM. Mark reported that CIMI is not ready for the level of detail needed for a true registry, but he will bring that up with them after the WGM (they are probably starting with a design repository at first, testing with one supplied by PortaVita). Kai will include HingX as an option for a templates registry in the upcoming discussions between the Art-Decor project and EPSOS.

### Mark presented a continuation of last week’s discussion on the analysis of template life cycles, introducing the idea of a ‘concept level’ template class which includes governance and purpose, and has a number of states, but where the template design(s) themselves are contained in related component-level ‘version(s)’ classes of that ‘concept level’ template as a way to consolidate the prior analysis on template states, template versioning, governance, etc. (see appendix A, below).

### Plans for next Tuesday’s conference call. Review statuses and updates on plans for the May WGM

## Appendix A. Concept level templates, and version level templates:

### Thinking about templates and their versions, from both a registry and a repository point of view.

### When one is creating a template, one is actually creating, (in v3-speak), an act (in event mood) that represents the template itself, plus a number of component acts (also in event mood), each of which represents a version of the template.

### The first act, the template itself, can be thought of as the ‘concept level template.’ It has a template ID, and a status , and documents the governance, and purpose of the template; at its creation, this act is identified by just its id and start date… but there is no design work as yet. It has a status=*draft*, and the actual design(s) are implemented in the ‘component classes’ or ‘version(s)’ of the of the ‘concept level’ template.

### Jane noted that this is parallel to the Vocabulary WG’s work with “concept domains” and “value sets”. The ‘concept level’ template is similar to the ‘concept domain’, and the versions (of that ‘concept level’ template) are similar to the versions of the value sets associated with a ‘concept domain’.

### Jane noted that this is a common mode of handling this type of complexity, and that it would be useful for not just CDA templates, but patient care templates as well.

### When the first design (version ) of the *concept level* template is created, it (as a component act <or quivalent> is identified by the template id (see ‘2’ above), the version number (‘1’ for the first version), and its own starting date and status (draft)

### the version may go through a number of states beyond draft, including: published (with a start date and status of active), return to draft (same version), ending (of several types: going to next version, no further versions (several types)). (See last week’s minutes.)

### 5. if the ending of the version is of a ‘no further versions’ category, the original ‘concept level’ template (see ‘2’ above) has one of several ending status: historical validity (no further use); historical not-valid for computable processing; and so on. (See last week’s minutes.)

### In the discussion, we also noted that the above lends itself to a clear conception of dealing with the complexity of template registry issues, since the registry needs to have both of the following accessible across user communities.

### the ‘concept-level’ template (with id, date(s) and status)

### and

### the version(s) (each with id (same as the concept-level), but also having a distinct version number, and version-level status and dates)

### where the ‘last version’ state & date (see ‘I.5’ above) may also change the status of the corresponding concept-level template (see last week’s minutes).