TMF compared to LMF
(auxiliary working paper for LMF)

Gil Francopoulo INRIA

30th July 2005

1 Introduction

TMF is a result of a nice job. But, for LMF, I think we have to take care and avoid to blindly mimic TMF.

It's not possible to follow a 100% identical strategy. There are numerous common points but there are also a certain number of differences. And these differences are not caused by what we want to do but instead are imposed both by the target and by some external constraints.

As a convention, I'll try to avoid the term "implementation"[1]. Instead, I prefer a more neutral term like "user defined model".

2 Common points

2.1 Common point #1: meta-model

Both standards are meta-models in the sense that:

- An abstract model is specified. This model describes the main structural elements together with their internal linking. The specification comprises a formal portion associated with a usage guideline for each element and linking. The usage guideline enables the user to avoid any misunderstanding. For instance, *Sense class* is intended to represent meaning and on the contrary *Inflectional Paradigm* is for paradigms not for meanings.

- This model is to be decorated by data categories taken from a data category selection (DCS).

- This model it to be mapped to a user defined model.

The standard specifies the meta-model and the mechanism to combine with DCS.

The standard does not define a user defined model. As a consequence and due to the fact that to define a mapping, three notions are required: source, rule of mapping and target; the mapping cannot be defined within the standard. Nevertheless some advices can be given, for instance, use XSLT or schema inclusion.

Let's note that the mapping can be a one-to-one mapping or a data-loss transformation. And considering the set of effective usages, the more one-to-one mappings are applied, the better the standard is.

---

[1] "implementation" is used today in so different meanings that it's no more possible to figure out what it is. Is it reification (transformation of something virtual into something concrete)? Is it deployment? Is it coding according to specifications? Is it the result of applying rules to a source model? Is it specialization from a generic pattern?

To conclude, the purpose of the meta-model is to act as a reference with regards to interoperable requirements.

2.2 Common point #2: best practices

Both models are based on best practices of their corresponding fields.

3 Differences

3.1 Difference #1: scope

The main difference is that obviously the scope addressed by LMF is much broader than TMF according to the following aspects:

- **Covered data types.** Lexicons as physical objects (in the real life) embrace a huge amount of linguistic data. By comparison, terminological resources scope is without any doubt much narrower. Just think of the data required by MT lexicons or CLIR smart indexing. And the data is of different nature: morphology, syntax etc.

- **Covered languages.** Once you enter into multi-lingual considerations, you need to cope with script coding, transcription forms etc.

- **Methodological short cut.** A strong methodological presupposition is implicit within TMF that is: in a specific technical domain, a term is translated through a language section node and reaches an **interlingual concept**. In other words, the interlingual hypothesis is taken for grant. This hypothesis has a strong consequence in terms of structural simplification. But this hypothesis goes up in smoke as soon as you go out of a specific technical domain, as lexicon use to do. Simply consult the "fleuve" / "river" example in LMF. Let's recall that lexicons address every day language and it's proven that this task is much more difficult than to tackle one specialized language.

The scope being broader, the specifications are more complex.

3.2 Difference #2: time

TMF is based on DTD and XSLT style sheets. **Modularity is non-existing**. XSLT becomes rapidly very complex and it's not very different from program writing.

Now the time is more in favor of schemas. Let's look around us. This year, OLIF replaced DTDs with schemas. TEI-P5 relies on schemas now. And the same technology shift can be noticed inside W3C, see for instance XHTML-2.0 or RDF. It's not a matter of fashion: it's merely because schemas are stable[2] and are technically more powerful.

LMF will be published somewhere around 2007. The next revision will be in 2012. We have to take decisions for the future not for the past.

---

[2] The situation is not so simple. In fact there are two rival technologies: Relax NG schemas and W3C schemas. Recognized both by ISO and W3C, Relax NG is more stable thanks to James Clarke associated with a strong Japanese group for the formal work. By comparison, during these last years W3C schemas seem to be stuck in the mood.

4 Conclusion

Concerning LMF, the task is not easy. The scope is broad.

We have to find something between the two undesirable extremes:

- One is to take a particular option specific to a certain school without any way to cope with options taken by other schools.

- One is to take a too simplistic strategy at the expense of a poor power of representation. Contrarily to what is often said, it's not because it's simple that it is powerful. The two notions (power and complexity) are orthogonal. The danger is great to propose a too naïve solution (of course, it's simpler to write) and get something known as: '**LMF is lexicon for dummies**'.

I'm not against simplicity. I'm against missing the target.

My opinion is that we must use modern technologies to cope with existing real lexicons and to take into account the user requirements for the future.

In this context that's why it is so essential to get foreseen users comments alongside LMF specification writing.