fhiso

CFPS 16
(Call for Papers Submission number 16)

Requirement for Locale Independence in Data Representations

Submitted by: Proctor, Tony

Created: 2013-03-11

URL: Most recent version: http://fhiso.org/files/cfp/cfps16.pdf
This version: http://fhiso.org/files/cfp/cfps16_v1-0.pdf

Description: Functional requirement for locale independence in representations of the Data Model.

Keywords: Locales, Portability, Serialisation-format

Contents

1. Abstract

Functional requirement that all representations of the Data Model (i.e. serialisation formats, including file formats) should be locale independent. This means that they should load identically in any other locale around the world.

2. Requirement

Locale Independence means that the interpretation of the data should not be dependent upon the locale setting of the end-user. It is primarily a computer issue because the data format is designed to be computer-readable. It is therefore a problem that has been solved already and there are applicable standards and best-practices for it.

2.1 Data Values

A good example of a locale-dependent pitfall is when you want to store a decimal value. Suppose your program stores the value exactly as your own locale expresses such values. A value stored as “3.14” in the US would be stored as “3,14” in France. When that same program reads the data back, everything seems to work OK in the US and in France. If the data files are exchanged, though, then the program will fail in both locations.

This problem affects any datum that is interpreted by the computer software when the data is loaded up, and that usually means numbers, dates & times, and Boolean values.

This subject has a direct parallel in the source code for programming languages. The source code is supposed to unambiguously define the actions of the program, irrespective of where it gets compiled into machine code. Hence, keywords should never be translated to different languages, and numeric, date/time, and Boolean constants should be in a fixed format that’s independent of the programmer’s locale setting. As a result, this fixed computer-readable format is often said to be belong to the ‘programming locale’.

2.2 Tags and Tag-Values

We take it for granted that the record types (e.g. element and attribute names in XML) are part of the computer-readable syntax and should never be translated into other languages.

There is a grey area, though, in textual values associated with some meta-data and data. For instance:

```
<Role Type=”Spouse”/>
<Sex>Male</Sex>
```

If such tag values are defined by the Data Model (i.e. part of its grammar) then they should never be translated into different languages. This is sometimes called a “[partially] controlled vocabulary”. Also, those tag values from the data should never be shown in the UI as they belong to the ‘programming locale’ and not the user’s locale. In those circumstances, the values should be mapped to locale-specific terms or descriptions applicable to the current end-user.

2.3 Sources and Citations

Traditional printed citations (e.g. CMOS) are inappropriate for locale independence. In additional to them not being digestible by software, there will be differences in punctuation and in numeric/date representation that will make them locale specific.

This has important repercussions for a Data Model. The representation of citations must be in a pre-digested form such as a collection of element values (e.g. author, publication date) and a reference to a registered citation form (e.g. a book, a Web page). It is not only far easier to generate readable citations from a digested form – as opposed to vice versa – it is also

possible to apply the necessary formatting for the regional and cultural preferences of the current end-user. This may include the printed citation style of which there are several alternative systems.

2.4 Character sets

A user's machine may have a default character set configured. The characters in a representation of the Data Model should be in a known character set and not allowed to be interpreted according to some indeterminate default. This means either adopting a fixed globally-applicable set, such as UTF-8, or having a character set identifier in the header. This situation is identical to that of XML.

Whether a serialisation format accommodates escaped references to characters – as per XML character entities and entity references – is a related issue that would not be mandatory but would be convenient.

3. Not Covered or Not Required

This requirement does not relate to textual data being in some specific language. The format is free to hold such data from many languages as long it is qualifies each one.

4. Illustration

This illustration uses the STEMMA syntax for properties (i.e. data values obtained from a specific source) to record a fractional age.

```
<Person Key='pKatherineSmith'>

    <EventRef Key='eCensusElliott1851'>
        <Property Name='Name'> Kate Smith </Property>

        <Property Name='Age' Units='m'> 2.5 </Property>
        <Property Name='BirthPlaceRef' Key='pUttoxeter'/>
        <Property Name='Role'> Daughter </Property>
    </EventRef>

</Person>
```

This records the census age of Kate Smith as 2.5 months. If the serialisation format did not stipulate the numeric format then this might be been represented as "2,5" is some other countries, and thus result in a locale dependency.

NB: The computer-readable format has no bearing on how it is displayed to the end-user. Just as with dates and times, this numeric value would be displayed accord to the end-user's regional settings and preferences.

5. Use Cases

Locale independence is necessary to ensure transportability of data files between different parts of the world and between users who may have configured different regional preferences.

6. References

STEMMA Locale Independence. http://www.familyhistorydata.parallaxview.co/home/locale-independence.

STEMMA research on world variations.
http://www.familyhistorydata.parallaxview.co/research-notes/worldwide-fh-data (Software Concepts and Standards section).