XLIFF in Practice

A view from the training and support side

Examples

File provided by client, when questioned, client said that alt-trans contained already known translations. If translation was available, the target did not need to be translated. Attribute state=translated helped to hide these segments.

```
<trans-unit id="1" extradata="unbekannt">
      <source>Abwasser</source>
      <target state="translated">Abwasser</target>
      <alt-trans>
        <target>Waste water</target>
      </alt-trans>
    </trans-unit>
```

Examples

With no languages set, the file cannot be imported into a tool for translation.

source-language="**default**" target-language="**default**"

Client wanted the translation in the note, not in the target area.

<trans-unit approved="yes" datatype="html"
id="additional_information.link" reformat="yes">
<source xml:lang="default">For more information</source>
<target state="new" xml:lang="default">For more information</target>
<note from="translator">For more information</note>
</trans-unit>

Examples

source language in header = default
source language in segments en-US
target language only in segments fi-FI

<file category="user" datatype="html" original="user"
source-language="default">
<header/>
<body>
<trans-unit approved="yes" datatype="html"
id="address.button.back.title" reformat="yes">
<source xml:lang="en-US">Go back to the user
overview.</source>
<target state="new" xml:lang="fi-FI"/>
<note from="creator">Go back to the user
overview.</note>

Examples

Some tools will not accept perfectly valid files according to the XML and/or XLIFF standards. For example, self-closing tags are accepted by some tools, whereas if the same tag is explicitly opened and closed, it won't work (probably because the tool is only designed to handle a self-closing tag). Others may have issues with spaces before the ending />.

```
<trans-unit id="100">
      <source>Unbekannt</source>
      <target />
    </trans-unit>

<trans-unit id="101">
      <source>Unbekannt</source>
      <target></target>
    </trans-unit>
```

Examples

HTML tags as entities within the text – tool would need a separate filter to be able to show these parts as tags.

<trans-unit id="99">
<source>
Unbekannt
</source>
<target>
Unbekannt
</target>
</trans-unit>

Examples

Invalid XML characters within the text, which most XML parsers will not accept. The tool will need to either strip those (which can lead to minor data loss) or escape them some way.

<trans-unit id=“100">
<source>Unbekannt&#11;</source>
<target>Unbekannt&#11;</target>
</trans-unit>

Issues

- Developers send XLIFF files, but the file extension is XML -> some tools will use the wrong filter to import the content for translation others are clever enough to give the info that there are several filters (XML, XLIFF…) available
- Source tag with source text is present, but some tools need also the target tag with a copy of the source text to be able to use it for translation
- XLIFF files come in a mixed state – some segments are already translated, others are not, but there is no attribute (like translate=true) that could be used to hide already translated text and show only new text.

Issues

- XLIFF files created by translation tools contain a lot of tool-specific information (status, translation match origin…) which at the moment cannot be re-used by any other tool. So what is the standard good for?
- Developers send XLIFF files with 40 MB – no tool will be able to open these.
- Developers specify source and target language with each segment pair instead of in the header

Solutions?

- Stricter rules on how to write XLIFF
- Status information should be re-usable between tools
- Training for developers -> what will the translation tools do with my XLIFF?