The translation process for 3R will include several steps that are entirely new. The new process is intended to make the work of translators easier and more efficient, but this initial translation process will be challenging as we work our way through the process and identify the problem areas. This document is meant to familiarize translators with the necessary steps for completing a translation and getting it published in the new Toolkit. The steps listed here suggest a schedule, but we are not in a good position to set a hard schedule, given the inevitable issues we will encounter in this initial translation and the varying levels of time and resources that each translation team has available.

Steps to completing the translation

1. **Translate RDA Reference** -- This is already underway. It includes translation of RDA Value Vocabularies and Element Sets. Ideally, teams will be close to completing this work in June, but again this is not a requirement.

2. **Trados Training** -- This too is underway for Daniel and James. We will be implementing new software for the RDA translation work. SDL Trados GroupShare is another SDL product that serves as a project management system for SDL Trados Studio. It will allow translation teams to better organize their work and share translation memories and term bases among themselves. In June, there will be GroupShare training for translation project managers, followed soon after by Studio training sessions for translators. There will 3 translator sessions in order to keep the groups small. The groups are 1) French, Spanish, Italian and Catalan; 2) German, Norwegian, Finnish and Hungarian; and 3) Arabic. All training sessions will be recorded and made available to the translations teams.

3. **Translate the "Translation Table"** -- This table is located in the Content Management System (CMS) and is a simple table that requires translation of repeated section and subsection headings associated with Entity and Element files (e.g., Definition and Scope, Element Reference, etc.). This table will be used in the generation of Entity/Element files. This step will be preceded by a CMS training session.

4. **Configure CMS directory, files, and ditamaps for each language** -- English language files and ditamaps, including the keydef map will be copied into each language folder. This will ensure the parallel structure of the directories and the transfer of the appropriate IDs. ALA Publishing will execute this step in June.

5. **Translate boilerplate file** -- “Boilerplate” refers to sentences and paragraphs that are frequently re-used in the RDA text. Boilerplate text is found in the bp_master file in the CMS and is the first file that should be uploaded to Studio for translation. The boilerplate is pulled into RDA instructions using content references or content key references, commonly referred to as conrefs or conkeyrefs (see the conref and key sections below for more on how this works).

6. **Run Operational Script** -- The Operational script pulls in the RDA Reference translation from the Registry and populates the Entity/Element files copied in the language folders (see section 4 above) with the Definition and Scope, Element Reference, and Related Elements sections. ALA Publishing will execute this step.
    a. The first run of the operational script will also insert the translated headings from the Translation table into the Entity/Element files (see section 3 above).
    b. The script, in combination with the translated boilerplate file, will automatically complete the translation of over 2,500 files that are composed of only registry data and boilerplate.

7. **StudioTranslation** -- The 600+ remaining files of RDA content will need to be translated in Studio. ALA Publishing will create the packets for translation and provide them to the translation teams. The likely flow of files will be as follows:
    a. Entity Chapters -- the files will flow in packets for each entity
    b. Guidance Chapters -- a single packet
    c. Related Resources -- a single packet.

8. **CMS editing** -- After Trados translation is complete the output will be uploaded to the CMS by ALA Publishing. Files should be reviewed in the CMS to check that they have processed properly. Any needed edits discovered in the review should be made in Studio, processed and uploaded anew to the CMS. This step will include further CMS training.

9. **User Interface** -- Translation of the Toolkit user interface (UI) is required before the publication of the translation on the Toolkit. ALA Publishing will provide a .po file for the translation of the UI text strings. The completed UI translation file should be returned to ALA Publishing.

10. **Examples** -- In the new toolkit, the examples are kept in a separate directory and conkeyreffed into the instructions. This allows the RDA Examples Editor to use examples in multiple places and gives the examples editor more control over the examples. The translation of English language examples and the development of unique examples for translations will be a bit trickier. A separate document on examples will be forthcoming. While examples are an important part of RDA Toolkit, they are not officially part of RDA, and they are expected to follow after the translation of the RDA text.

11. **Help** -- This content is not required by the translation agreement, but teams are welcome to translate it. ALA Publishing will provide files, and translation of the material should be done inStudio. Screen shots used in the Help section are from the English version. Screen shots from the appropriate language version can be added to a Help translation once the RDA translation is completed.

Important concepts

There are several important concepts associated with DITA and Studio that are critical to the architecture of RDA and are frequently used and discussed in the production process. The

translators should familiarize themselves with these topics to better understand the translation process.

Content references
The content reference, commonly called conref, is a kind of link in reverse. A regular link takes you from the document you are in to a new piece of text in a different document. A conref takes a piece of text in one document and places it in the document you are in. This allows editors to re-use content from another DITA file in a file they are editing. In RDA, this most frequently involves re-using boilerplate text within multiple element files, although this is also used to insert examples into RDA instructions.

There are 2 advantages to the conref. First, it ensures consistency of wording. Second, if you need to change that wording, you only need to do it once in the boilerplate file and the change is then reflected in the other files.

- Conref content appears in the CMS editor tool with a grey background.
- Conref content will not appear in Studio.

A content key reference, commonly called a conkeyref, uses a key in the link rather than a filename. Conkeyrefs are the preferred link in RDA over conrefs and the more frequently used.

DITA maps
The DITA map is a means for organizing DITA files and subunits into specific order. DITA maps are a distinct file type (.ditamap). RDA has two primary uses for DITA maps:

1. Pulldown menu entries (for example, work_menu.ditamap), and

2. Page browsing/controlling the navigation buttons (previous/next) at the foot of element pages (for example, work_browse.ditamap).

It should be noted that DITA maps can include other DITA maps, and you may open a DITA file from a DITA map.

Translation teams will need to review DITA maps and edit the maps so that they are properly alphabetized for each language.

Keys
Keys are persistent identifiers for files and are defined in a special kind of ditamap called a keydef map. The keys are used to create links using keyrefs and conkeyrefs. The key can replace the target filename in a link, so if the filename changes or the target of the link changes, the editor can simply revise the keydef map and none of the links will break.

Tags and their attributes
The tags found in RDA files are generally familiar to those who have worked in XML environment. The typical file uses the following structural tags in a hierarchical structure. The asterisk indicates required tags.

- Topic*
- Body*

- Section*
- Division (div)
- Paragraph (p)

Tags have attributes. Most importantly, these structural tags all have an ID attribute. These IDs are critical to RDA Toolkit supporting translations. The IDs must be consistent throughout the translations in order for the site to work.

The other important and frequently-used attribute is outputclass. This is used to indicate specific display requirements. In RDA the output class is used to signal header, condition, and option formatting.

Finally there are some links in RDA to external websites that reference other international standards. This links will need to be reviewed as the translator may want to redirect the link to a language specific version of the link target. FOr example a link to an IFLA document in English may be revised to point to the French or Arabic version of the document.

Translations memories

Most of you should be familiar with translations memories in Studio. This will be essential to the editing and maintenance of your translation. The CMS files and the translation memory need to match. If you make a change in the CMS, it will not get picked up in the translation memory and the next export from Trados will overwrite the changes. To prevent this from happening, any text edits need to be made in Trados. This will update the translation memory and then the export from Studio can be uploaded to the CMS.