TBX-Basic Implementation Guide

Validation – Import – Export

Contents

Introduction

TBX-Basic is intended to be the primary dialect for terminology exchange. It can be used to handle monolingual, bilingual, or multilingual glossaries or termbases. It is composed of three modules: Core, Min, and Basic. The Core module consists of the core structure which all valid TBX files share and which corresponds with the description of the TBX Core as found in ISO 30042. The combination of the data categories added to the Core by the Min and Basic modules constitutes the TBX-Basic dialect. This implementation guide is provided by LTAC Global as a public service. If you want to implement TBX-Basic, you are free to use it. If you want to create your own TBX dialect, you must obtain appropriate access to a purchased copy of the new version of ISO 30042 DIS (to be published in 2018).

Validation

This section details the steps which must be taken to ensure an XML document instance (hereafter described as the "*candidate*") which claims to be TBX-Basic is actually a valid instance of the TBX-Basic dialect. If at any step the *candidate* fails a test, it is not valid TBX-Basic.

If the *candidate* does not even claim to be compliant with TBX-Basic (according to the type attribute on the root), use the Validation API to attempt to learn which modules the *candidate's* stated dialect is supposed to contain.

TBX-Basic Validation Steps:

1. Check if *candidate* looks like XML.
2. Ensure the *candidate* is well-formed XML.
3. The root start-tag should declare the default namespace as: urn:iso:std:iso:30042:ed:3.0
    a. Example:
    <tbx type="TBX-Basic" style="dca" xml:lang="en" xmlns="urn:iso:std:iso:30042:ed:3.0">
4. The root start-tag must contain a @type attribute, the value of which must be "TBX-Basic".
    a. Example:
    <tbx type="TBX-Basic" style="dca" xml:lang="en" xmlns="urn:iso:std:iso:30042:ed:3.0">

5. The root start-tag must contain a @style attribute, the value of which must be either “dca” or “dct”. [1] The next step of validation depends on the claimed style of TBX.
    a. Example:
    <tbx type="TBX-Basic" style="dca" xml:lang="en" xmlns="urn:iso:std:iso:30042:ed:3.0">

Using any off-the-shelf XML validator that supports the RelaxNG and Schematron languages (or whatever languages your equivalent schemas are written in), the schemas discussed below can be used to finish the validation of the *candidate*.

IMPORTANT NOTE: In practice (in software such as Oxygen), it is possible to simply point to the URL of these schemas (as found in the footnotes) directly from a TBX file, bypassing the need for a local file entirely. For DCT, it is only necessary point to the NVDL schema, as it ties the other DCT schemas together.

Note: For some validation applications, such as a tool which helps design a TBX dialect, it is permissible to use the RNG for the Core module (see Schemas for DCT style TBX for link) for partial validation. It will show whether a TBX file adheres to the core structure of TBX. However, this partial validation is NOT considered acceptable for Import/Export routines or any other validation routine which needs to validate a specific instance of a TBX dialect.

Schemas for DCA style TBX

The following resources are needed to validate a TBX-Basic file of DCA style:

- TBX-Basic integrated RelaxNG schema (or equivalent).[2]
    - The TBX-Basic RelaxNG schema, found on TBXinfo.net[3] is a modified version of the TBX core RelaxNG schema (see DCT section for link). Using existing extension points which are shown on TBXinfo.net, the core schema was modified for use by TBX-Basic files.
- TBX-Basic DCA style Schematron schema (or equivalent).[4]

Schemas for DCT style TBX

The following resources are needed to validate a TBX-Basic file of DCT Style:

- Core Module RNG schema (or equivalent) [5]

---

[1] These acronyms stand for “Data Category as Attribute” and “Data Category as Tag” (aka, general identifier). Examples of both styles of TBX using the “subjectField” data category:

DCA -> <descrip type=”subjectField”>law</descrip>
DCT -> <basic:subjectField metaType*=”descrip”>law</basic:subjectField>**

*The @metaType is optional and helps DCT and DCA remain isomorphic, capable of lossless conversion from one style to the other.
**Unlike DCA style, DCT imports data categories from modules via namespace. This example assumes that the “basic” namespace prefix has been defined in the <tbx> root element.

[2] https://raw.githubusercontent.com/LTAC-Global/TBX-Basic_dialect/master/DCA/TBXcoreStructV03_TBX-Basic_integrated.rng

[3] http://www.tbxinfo.net/ [note: for now, consult the development website: http://tbxinfo-dev.byuling.net/]

[4] https://raw.githubusercontent.com/LTAC-Global/TBX-Basic_dialect/master/DCA/TBX-Basic_DCA.sch

[5] https://raw.githubusercontent.com/LTAC-Global/TBX_Core_RNG/master/TBXcoreStructV03.rng

- Min module RNG (or equivalent)[6]
- Min module SCH (or equivalent)[7]
- Basic module RNG (or equivalent)[8]
- Basic module SCH (or equivalent)[9]
- TBX-Basic DCT style Schematron schema (or equivalent)[10]
- TBX-Basic NVDL schema (or equivalent)[11]

Validation API

A simple TBX validation RESTful API has been created and can be accessed at http://validate.tbxinfo.net/. It can be used to programmatically get the absolute URLs to the various schemas of dialects and modules, which can then be downloaded and parsed (or pointed to directly). It can also be used to get a list of available modules, or the list of modules which are used to define a dialect. The response from the API is in JSON format.

= = = [Not yet written: how to develop a TBX-Basic import routine; same for export] = = =

Notes for the import export sections, to be written:

- A TBX-Basic import routine consumes a TBX-Basic document instance and inserts information into a termbase. This guide does not cover how to deal with duplicate entries (same term as an entry already in the database, which may or may not designate the same concept as the existing entry).
- A TBX-Basic export routine consumes a list of concept entries in an existing termbase and represents the information in them as a valid TBX-Basic document instance, mapping from the data categories in the termbase to TBX-Basic data categories as needed.

[6] https://raw.githubusercontent.com/LTAC-Global/TBX_min_module/master/Min.rng
[7] https://raw.githubusercontent.com/LTAC-Global/TBX_min_module/master/Min.sch
[8] https://raw.githubusercontent.com/LTAC-Global/TBX_basic_module/master/Basic.rng
[9] https://raw.githubusercontent.com/LTAC-Global/TBX_basic_module/master/Basic.sch
[10] https://raw.githubusercontent.com/LTAC-Global/TBX-Basic_dialect/master/DCT/TBX-Basic_DCT.sch
[11] https://raw.githubusercontent.com/LTAC-Global/TBX-Basic_dialect/master/DCT/TBX-Basic.nvdl

API Example for TBX-Basic

URL: http://validate.tbxinfo.net/dialects/TBX-Basic

Response:


```
[{
   "name": "TBX-Basic",
   "definition": "https://github.com/LTAC-Global/TBX-Basic_dialect/blob/master/TBX-
Basic%20Definition.pdf",
   "dca_rng": "https://raw.githubusercontent.com/LTAC-Global/TBX-
Basic_dialect/master/DCA/TBXcoreStructV03_TBX-Basic_integrated.rng",
   "dca_sch": "https://raw.githubusercontent.com/LTAC-Global/TBX-Basic_dialect/master/DCA/TBX-
Basic_DCA.sch",
   "dct_nvdl": "https://raw.githubusercontent.com/LTAC-Global/TBX-Basic_dialect/master/DCT/TBX-
Basic.nvdl",
   "dct_sch": "https://raw.githubusercontent.com/LTAC-Global/TBX-Basic_dialect/master/DCT/TBX-
Basic_DCT.sch",
   "id": 2,
   "modules": [
      {
         "name": "Min",
         "definition": "https://github.com/LTAC-
Global/TBX_min_module/blob/master/Min%20Module%20Definition.pdf",
         "rng": "https://raw.githubusercontent.com/LTAC-Global/TBX_min_module/master/Min.rng",
         "sch": "https://raw.githubusercontent.com/LTAC-Global/TBX_min_module/master/Min.sch",
         "tbxmd": "https://raw.githubusercontent.com/LTAC-
Global/TBX_min_module/master/Min.tbxmd",
         "id": 1
      },
      {
         "name": "Basic",
         "definition": "https://github.com/LTAC-
Global/TBX_basic_module/blob/master/Basic%20Module%20Definition.pdf",
         "rng": "https://raw.githubusercontent.com/LTAC-Global/TBX_basic_module/master/Basic.rng",
         "sch": "https://raw.githubusercontent.com/LTAC-Global/TBX_basic_module/master/Basic.sch",
         "tbxmd": "https://raw.githubusercontent.com/LTAC-
Global/TBX_basic_module/master/Basic.tbxmd",
         "id": 2
      },
      {
         "name": "Core",
         "definition": "",
         "rng": "https://raw.githubusercontent.com/LTAC-
Global/TBX_Core_RNG/master/TBXcoreStructV03.rng",
         "sch": "",
         "tbxmd": "",
         "id": 4
      }
   ]
}]
```

Using the response:

For DCA, only the schemas pointed to by "dca_rng" and "dca_sch" are needed. For DCT, the schemas pointed to by "dct_nvdl", "dct_sch", and each of the "rng" and "sch" schemas for each module of the "modules" list. As mentioned before in the DCT section, it is only necessary in practice to use the "dct_nvdl" schema link, since by default, it points to the absolute URLs of the other schemas needed in DCT.

Import

This section details the process of importing a TBX file.

1. Validate the file to be imported according to the instructions from the Validation section.
    a. If the file fails validation, it should be rejected.
    b. Providing a report of some kind indicating the reason for failure is recommended.
2. Using the Validation API or TBXinfo.net, retrieve the list of modules for the dialect of the file (TBX-Basic will be used here for demonstration).
    a. Using either the prose definition, the RNG and SCH schemas (or equivalent), or the TBXMD file for each module, you can learn exactly which data categories to expect on import. The TBXMD file would be the simplest to parse to learn this information. The description of the TBXMD file can be found on the TBXinfo.net.[12]
    b. For TBX-Basic these modules will be: Core, Min, and Basic.
3. If any module is unsupported by the importing software, you may:
    a. omit them with a message that the data categories from said module are unsupported;
    b. convert them to /note/ data categories via a converter.
        i. It is recommended that the original data category name be preserved as a part of the note, so that conversion back to the appropriate data category at some future point is made possible.
4. Any data categories from modules which are supported may be imported normally.

Export

This section details the process of exporting a file instance of a certain dialect (TBX-Basic in this case).

1. Select the dialect to which you will be exporting. (TBX-Basic)
    a. **Note: Generic exports of "TBX" files without a clearly stated dialect are not valid.**
2. Decide which style of TBX the export which be (DCA or DCT).
    a. Mixing styles is not permitted.
3. Use the Validation API or TBXinfo.net to retrieve the list of modules included by the target dialect.
    a. TBX-Basic includes: Core, Min, and Basic
4. If the data model of the exporting software already maps to the data categories included by the modules, a simple export should suffice.
5. If the data model of the exporting software does not directly map to the data categories included by the module, a manual mapping wizard may be necessary.
    a. If only some of the modules of the target dialect are supported, the unsupported data categories may either be converted to /note/s or omitted (preferably with a report to the user).

[12] http://tbxinfo-dev.byuling.net/wp-content/uploads/2017/11/TBX%20Module%20Description.pdf

6. In either situation, the output TBX file must use the data category names which are defined by the modules for that dialect and not the internal names for said data categories.
    a. Example: Internally, the software may call /subjectField/ “domain”, but on export it must be converted to “subjectField”.
    b. This same rule applies to defined picklist values. If the software uses the abbreviated “adj” internally, on export, it must be changed to match whatever values are declared in the module definition. In the case of TBX-Basic, “adj” would need to be changed to “adjective”.
7. The export file must be a well-formed XML file with a “.tbx” file extension
8. The export file must also be valid instance of the target dialect (TBX-Basic) according to the schemas of that dialect and style (DCA or DCT).