This document is part of the Coordination and Support Action “Preparation and Launch of a Large-scale Action for Quality Translation Technology (QTLaunchPad)”. This project has received funding from the European Union’s Seventh Framework Programme for research, technological development and demonstration under grant agreement no. 296347.

Deliverable D 1.4.1

TQ Test Suite

Author(s): Arle Lommel,
Aljoscha Burchardt
Kim Harris (text&form)
Maja Popović
Hans Uszkoreit

Dissemination Level: Public

Date: 20.November 2014 (revision)

Grant agreement no.	296347
Project acronym	QTLaunchPad
Project full title	Preparation and Launch of a Large-scale Action for Quality Translation Technology
Funding scheme	Coordination and Support Action
Coordinator	Prof. Hans Uszkoreit (DFKI)
Start date, duration	1 July 2012, 24 months
Distribution	Public
Contractual date of delivery	February 2014
Actual date of delivery	February 2014 (revision November 2014)
Deliverable number	1.4.1
Deliverable title	TQ Test Suite
Type	Other
Status and version	Final, v3.0
Number of pages	
Contributing partners	DFKI
WP leader	DFKI
Task leader	DFKI
Authors	Arle Lommel, Aljoscha Burchardt, Kim Harris (text&form), Maja Popović, Hans Uszkoreit
EC project officer	Aleksandra Wesolowska
The partners in QTLaunchPad are:	Deutsches Forschungszentrum für Künstliche Intelligenz (DFKI), Germany Dublin City University (DCU), Ireland Institute for Language and Speech Processing, R.C. “Athena” (ILSP/ATHENA RC), Greece The University of Sheffield (USFD), United Kingdom

For copies of reports, updates on project activities and other QTLaunchPad-related information, contact:

DFKI GmbH
QTLaunchPad
Dr. Aljoscha Burchardt aljoscha.burchardt@dfki.de
Alt-Moabit 91c Phone: +49 (30) 23895-1838
10559 Berlin, Germany Fax: +49 (30) 23895-1810

Copies of reports and other material can also be accessed via http://www.qt21.eu/launchpad

Table of Contents

1 Introduction and Motivation 4

2 Description 5

2.1 Issue types 7

3 Data structure 8

4 Sorting and filtering 9

4.1 Sorting by table column 9

4.2 Filtering 9

5 License 11

6 Usage scenario and sustainability 11

7 Effort of building a test suite 12

8 XML data format 12

9 References 14

10 Changes from previous version 14

1 Introduction and Motivation

This Deliverable is the final point in a series of Deliverables that apply the Multidimensional Quality Metric (MQM) developed in the QTLaunchPad project (see Deliverable 1.1.2) to MT output in order to delimit the current barriers for high-quality Machine Translation (HQMT).

The first steps consisted in the creation of a translation quality error corpus (see Deliverable 1.2.1), and an analysis of those errors that occur in “almost perfect” translations, which are roughly 1-3 errors away from being perfect (see Deliverable 1.2.2). The third step resulted in a large “Barriers for HQMT” database documenting these types of errors (see Deliverable 1.3.1).

Test suites are a familiar tool in NLP e.g., in the area of grammar checking, where one may wish to ensure that a parser is able to analyse certain sentences correctly or test the parser once it has been changed to see if it still behaves in the expected way. By test suite, we refer to a selected set of input-output pairs that reflects interesting or difficult cases. In contrast to a corpus that includes reference translations like the QTLaunchPad error corpus (D1.3.1), the input in a test suite may well be made-up or edited to isolate and illustrate issues.

Test suites have not generally been used in machine translation (MT) research. One of the reasons for this might be the fear that the performance of statistical MT systems depends so much on the particular input data, parameter settings, etc. that final conclusions about the errors they make and in particular about the different reasons (e.g. length of n-grams, missing training examples) are difficult to obtain. We are nevertheless convinced that testing of system performance on error classes leads to insights that can guide future research and improvements of systems. The test suites presented in this Deliverable are pioneering work towards a systematic fight against translation barriers. By using test suites, MT developers will be able to see how their systems perform compared to scenarios that are likely to lead to failure and to take corrective action.

The QTLaunchPad description of work requested a total of 250 sentences for one language pair as a proof of concept. However, the consortium together with GALA and a subcontracted Language Service Provider (text&form) has undertaken to create a substantially larger test suite, covering both English→German and German→English. Rather than deliver this test suite as a one-time result, DFKI will continue to maintain and expand on the test suite, with plans to extend it with examples from other projects (e.g., QTLeap) and to add data to it on an ongoing basis to the advantage of the MT research community. The test suites can be found at the following URL: http://www.qt21.eu/deliverables/test-suites/

2 Description

This resource contains a table of machine-translated segments that show errors. The errors were evaluated and the segments selected by expert linguists and selected because they either represent a barrier for MT systems or because they exhibit errors that exemplify typical problems that arise in MT scenarios. The data were derived from two sources:

1. **Segments from QTLaunchPad corpora or other resources**. These segments were selected from annotated QTLaunchPad corpora[1] to illustrate particular issue types. In addition, segments from various other sources were selected to demonstrate common problems. In some cases sentences were generated to illustrate common problems with a minimal example.
2. **Segments from the TSNLP Grammar Test Suite**. To prepare this corpus all of the "grammatical" segments from TSNLP ([1]) for the appropriate source language were reviewed. As TSNLP was not designed for use in MT testing, but rather to provide challenging cases for grammar checkers, a team of two native-speaker linguists evaluated all segments for each language and only those segments that both reviewers agreed were truly grammatical and relevant for MT diagnosis were used. In addition, sentence fragments were removed since isolated sentence fragments pose particular problems even for human translators. The resulting set of sentences was then translated using four leading commercial MT systems (two SMT and two RbMT) and sentences that proved problematic for both systems of a given MT type were classified as exhibiting a barrier for that system type.

For the TSNLP data, one MT result was selected from among those systems that were considered to exhibit barriers. This segment was the one judged by the group of linguists to come the closest to "getting it right". For the corpus data, the translation in the corpus was used. In both cases the translation was annotated using MQM to identify issues and post-edited to show one possible way to resolve the issues. Note that the post-editing was intended to be minimal, with only enough changes to make the sentence grammatical and acceptable. Full post-editing in many cases would result in more substantive changes in sentence structure, but the goal was not to create a stylistically perfect text.

[1] http://www.qt21.eu/deliverables/annotations/

For the corpus data, no information is provided as to which system type translated the segment, for which system type(s) the segments proved to be a barrier, or the TSNLP class.

The test suite consists of two separate suites:

1. English→German Test Suite
   - http://www.qt21.eu/deliverables/test-suite/test-suite-en-de.html
   - As of December 2014, this suite consists of 361 TSNLP sentences and 127 corpus sentences.
2. German→English Test Suite
   - http://www.qt21.eu/deliverables/test-suite/test-suite-de-en.html
   - As of December 2014, this suite consists of 142 TNSLP sentences and 149 corpus sentences.[2]

The data is available in a downloadable XML format as well as through the web interface. The format is described in **Section 8**.

NOTE: Local copies of the test-suite data may not render properly without an active Internet connection as they require access to online stylesheets for rendering.

Considerable effort was put into this Deliverable. Over several weeks, regular calibration meetings were held and sample annotations were discussed to arrive at a consistent sampling and annotation scheme. All in all, six persons (German and English natives or near-natives) were involved in the sampling and annotation. Some of the data was double annotated or cross-checked.

We found that production of a test suite is a very labour-intensive task, with hundreds of hours required to identify candidates, analyse them, and provide suitable documentation, particularly in the WMT and customer data, where many issues might interact with each other and where the sentences exhibited a wide range of complex issues. By contrast, the TSNLP data provided compact examples that were generally straightforward to evaluate, but the exemplars have the disadvantage of being somewhat unnatural at times. By taking both data “from the wild” and the systematically created TSNLP examples, however, we were able to balance the strengths of SMT and RbMT systems: for the TSNLP data the RbMT

[2] In both cases the number of corpus sentences is expected to grow over time while the number of TSNLP sentences is likely to remain constant.

systems performed significantly better than the SMT systems, but for the customer and WMT data the SMT systems generally exhibited fewer barriers.

2.1 Issue types

The majority of data was annotated with the MQM issue types used in the second round of annotation, as described in D1.3.1. All training materials described in that deliverable apply to this task. The guidelines and decision tree used can be downloaded from http://qt21.eu/downloads/annotatorsGuidelines-2014-06-11.pdf.

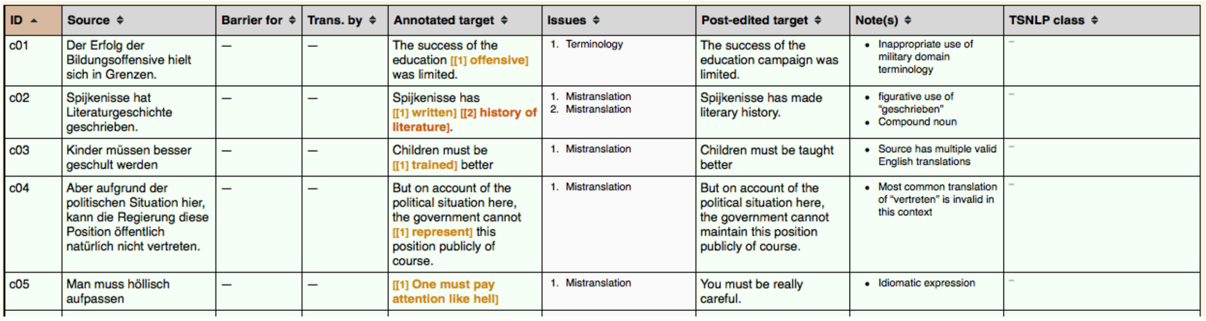
However, after the bulk of annotation was completed, the decision was made to augment the test suite with exemplars of other, more granular MQM types or additional MQM types that were not covered in the data annotated for D1.3.1. The full issue set, including these additional types, is visible in the online filtering options described in **Section 4**. The additional issues included are the following:

1. Terminology
2. Overly literal
3. False friend
4. Entity
5. Should not have been translated
6. Date/time
7. Unit conversion
8. Number
9. Register
10. Inconsistency
11. Unidiomatic
12. Capitalization
13. Punctuation
14. White space
15. Unpaired quote marks or brackets
16. Locale convention
17. Date format
18. Time format
19. Measurement format
20. Number format
21. Quote mark type
22. National language standard
23. Character encoding

These items are defined in the MQM online definition’s list of issue types[3].

3 Data structure

Figure 1 provides a screen shot of some rows from the German→English test suite.



ID	Source	Barrier for	Trans. by	Annotated target	Issues	Post-edited target	Note(s)	TSNLP class
c01	Der Erfolg der Bildungsoffensive hielt sich in Grenzen.	—	—	The success of the education [[1] offensive] was limited.	1. Terminology	The success of the education campaign was limited.	• Inappropriate use of military domain terminology	–
c02	Spijkenisse hat Literaturgeschichte geschrieben.	—	—	Spijkenisse has [[1] written] [[2] history of literature].	1. Mistranslation 2. Mistranslation	Spijkenisse has made literary history.	• figurative use of “geschrieben” • Compound noun	–
c03	Kinder müssen besser geschult werden	—	—	Children must be [[1] trained] better	1. Mistranslation	Children must be taught better	• Source has multiple valid English translations	–
c04	Aber aufgrund der politischen Situation hier, kann die Regierung diese Position öffentlich natürlich nicht vertreten.	—	—	But on account of the political situation here, the government cannot [[1] represent] this position publicly of course.	1. Mistranslation	But on account of the political situation here, the government cannot maintain this position publicly of course.	• Most common translation of “vertreten” is invalid in this context	–
c05	Man muss höllisch aufpassen	—	—	[[1] One must pay attention like hell]	1. Mistranslation	You must be really careful.	• Idiomatic expression	–

Figure 1. Screen shot from the online German→English test suite.

As can be seen in **Figure 1**, data in the test suite files is in the following columns:

1. **ID**. A unique ID value that can be used to identify an item in the test suite. For “corpus” segments, the ID value begins with the letter **c**; for items from the TSNLP suite, they begin with **g** and are followed by the ID value from the TSNLP suite.
2. **Source**. The source segment
3. **Barrier for**. Which system type(s) for which the segment is a barrier
4. **Trans. by**. Which system translated the segment that was annotated and post-edited. (SMT indicates a statistical system; RbMT a rule-based system.)
5. **Annotated target**. The target segment with markers for the annotated spans. (Note that MQM allows for overlapping, i.e., non-nested annotations, but all annotations in this test suite are strictly nested.)
6. **Issues**. A numbered list of issue types, corresponding to the marked spans in the Annotated target column.
7. **Post-edited target**. Contains a post-edited version of the target text. Specific changes from the original target are not marked, but can be easily determined by comparison of the target and post-edited target
8. **Notes**. Human-readable notes concerning the item.
9. **TSNLP class** (for TSNLP items only). The TSNLP class for the source segment.

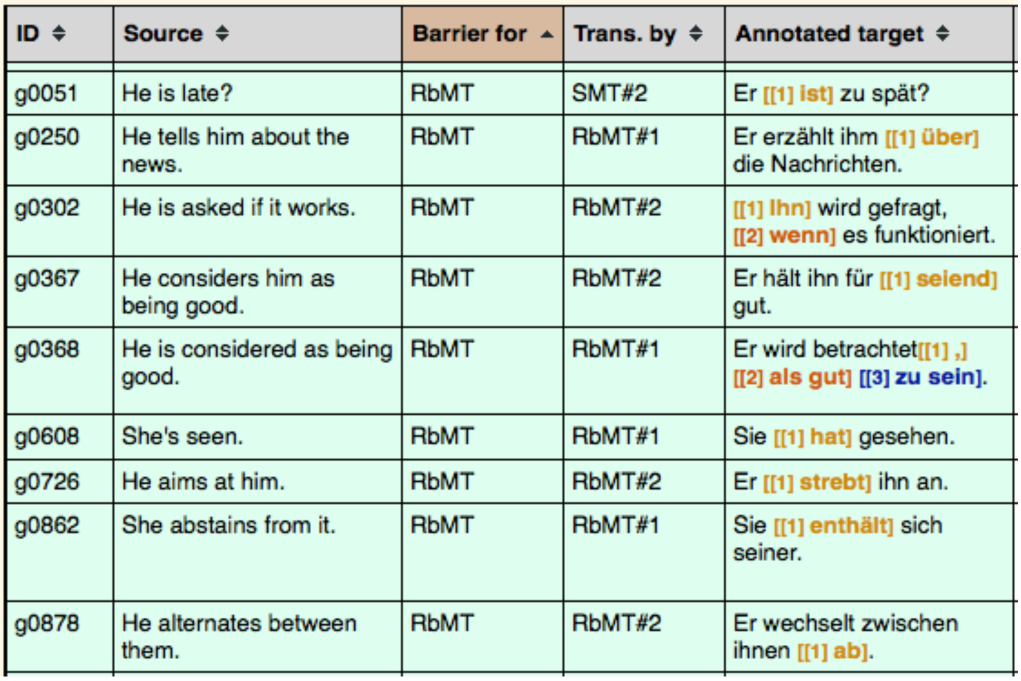
[3] http://www.qt21.eu/mqm-definition/issues-list.html

4 Sorting and filtering

The data sets provide options to filter data. Filtering data uses the JQuery JavaScript framework and requires an active Internet connection.

4.1 Sorting by table column

Clicking on a table column's heading will sort the contents of the table by the value of that column. Sorting is strict alphanumeric. Sorting is useful in finding, for example, all of the segments that were a barrier for a particular type of system. In **Figure 2**, for example, clicking on the heading *Barrier for* allows all segments that were a barrier for rule-based systems to be grouped together.

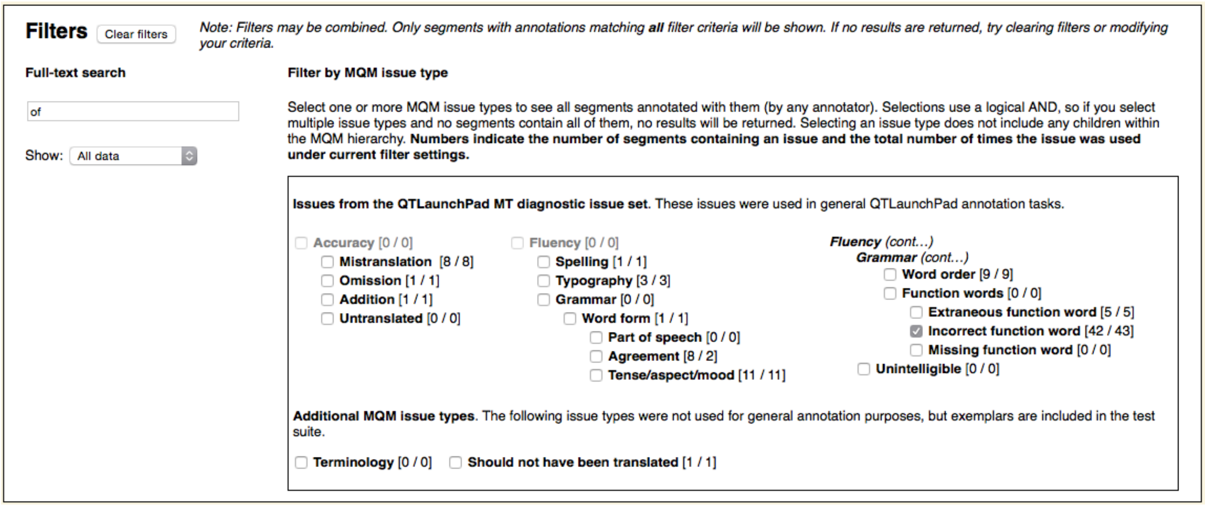


ID ⇕	Source ⇕	Barrier for ▲	Trans. by ⇕	Annotated target ⇕
g0051	He is late?	RbMT	SMT#2	Er [[1] ist] zu spät?
g0250	He tells him about the news.	RbMT	RbMT#1	Er erzählt ihm [[1] über] die Nachrichten.
g0302	He is asked if it works.	RbMT	RbMT#2	[[1] Ihn] wird gefragt, [[2] wenn] es funktioniert.
g0367	He considers him as being good.	RbMT	RbMT#2	Er hält ihn für [[1] seiend] gut.
g0368	He is considered as being good.	RbMT	RbMT#1	Er wird betrachtet[[1] ,] [[2] als gut] [[3] zu sein].
g0608	She's seen.	RbMT	RbMT#1	Sie [[1] hat] gesehen.
g0726	He aims at him.	RbMT	RbMT#2	Er [[1] strebt] ihn an.
g0862	She abstains from it.	RbMT	RbMT#1	Sie [[1] enthält] sich seiner.
g0878	He alternates between them.	RbMT	RbMT#2	Er wechselt zwischen ihnen [[1] ab].

Figure 2. Sorting the table to show segments that were a barrier for rule-based MT systems.

4.2 Filtering

The Test Suite contains a powerful set of filters that allow users to explore the data, as shown in **Figure 3**:



Filters Clear filters *Note: Filters may be combined. Only segments with annotations matching **all** filter criteria will be shown. If no results are returned, try clearing filters or modifying your criteria.*

Full-text search

of

Show: All data

Filter by MQM issue type

Select one or more MQM issue types to see all segments annotated with them (by any annotator). Selections use a logical AND, so if you select multiple issue types and no segments contain all of them, no results will be returned. Selecting an issue type does not include any children within the MQM hierarchy. **Numbers indicate the number of segments containing an issue and the total number of times the issue was used under current filter settings.**

Issues from the QTLaunchPad MT diagnostic issue set. These issues were used in general QTLaunchPad annotation tasks.

- □ Accuracy [0 / 0]
  - □ **Mistranslation** [8 / 8]
  - □ **Omission** [1 / 1]
  - □ **Addition** [1 / 1]
  - □ **Untranslated** [0 / 0]
- □ Fluency [0 / 0]
  - □ **Spelling** [1 / 1]
  - □ **Typography** [3 / 3]
  - □ **Grammar** [0 / 0]
    - □ **Word form** [1 / 1]
      - □ **Part of speech** [0 / 0]
      - □ **Agreement** [8 / 2]
      - □ **Tense/aspect/mood** [11 / 11]

***Fluency** (cont...)*

- ***Grammar** (cont...)*
  - □ **Word order** [9 / 9]
  - □ **Function words** [0 / 0]
    - □ **Extraneous function word** [5 / 5]
    - ☑ **Incorrect function word** [42 / 43]
    - □ **Missing function word** [0 / 0]
- □ **Unintelligible** [0 / 0]

Additional MQM issue types. The following issue types were not used for general annotation purposes, but exemplars are included in the test suite.

□ **Terminology** [0 / 0] □ **Should not have been translated** [1 / 1]

Figure 3. Filter options

The following filter options are available:

- **Full-text search**. Search addresses both source and target texts, as well as postedited segments. Search is case insensitive and ignores tag numbers displayed in the Test Suite (e.g., if a segment appears as “**[[1] Er]** **[[2] aller]** ist neu”, searching for “Er aller” will find the segment). Issue numbers in the text are not searchable.
- **Selecting which sort of data to show**. It is possible to select whether to see corpus data, TSNLP data, or both using the **Show** menu.
- **Filtering by MQM issue type**. Selecting an MQM issue type will reveal all segments annotated with that issue type. Multiple issue types may be selected simultaneously to see all segments containing all of the issue types (i.e., selections use a logical AND). The numbers in the interface show (a) how many segments contain each issue type under the current filter settings and (b) how many total times the issue type is used under the current filter settings. For example, if an issue type is followed by “[2 / 3]” it would indicate that the issue type occurs in two total segments under the current filter settings and that it appears three total time (i.e., one of the segments contains two annotations for that issue type).

Note that all filters may be combined. For example, it is possible to set up a filter to see all TSNLP data containing the word “of” and an Incorrect function word. Filters always interact using a logical AND.

To clear all filters and restore the display to its original state, click on the **Clear filters** button. Note that table sorting is not affected by filters.

5 License

All data contained in the QT Test Suite is available under a Creative Commons Attribution-ShareAlike 4.0 International License.

6 Usage scenario and sustainability

Test suites provide a set of sentences that can be used to determine whether an MT system is prone to produce specific error types. It can be used in different scenarios, e.g.:

1. **Generating research & development goals**: Either by inspection of the given illustrative MT output or output of another system, one can develop strategies for fixing the issues. The filtering options help to derive complex hypotheses.
2. **Automatic testing**: One can use the test suite (or a subset) as targeted reference corpus and use the existing distance-based measures to experiment with different system configurations and see if the respective system output gets closer to the reference translation or not. As with any other reference translation, if MT output gets closer, this is an *indication* for translation quality improvement (i.e., for having found a solution for the given issue). If not, this is an *indication* for a drop translation quality (i.e., for not having solved the issue). But today's automatic measures cannot provide a proof of improvement of translation quality (or solving an issue).
3. **Manual testing**: Once a system is considered to be in an interesting development stage, a human can quickly go through the output just answering "yes" or "no" to the question if the system (still) makes a particular error or if the issue has been solved.

To avoid the risk of overfitting, the test suite should not be used for development or system tuning. It is a means for testing. There is a risk of cheating as pointed out by the reviewers of QTLaunchPad. This risk pertains to any comparable benchmark that is know before testing. However, it is relatively straightforward to (semi-automatically) generate variants of the test items by lexical substitution, paraphrasing, etc. For benchmarking events, unseen versions of the test suites can be created with much less effort than it would take to build a completely new test suite.

Once test suites have established, they should be further developed over time as a community action. If all relevant projects would add test items to a shared repository, it would grow and at the same time cope with the state if the art.

DFKI has concrete plans to team up with translation departments of universities for generating more test items through the involvement of students.

7 Effort of building a test suite

Building test items comprises several steps. To add an item, we must:

1. Obtain multiple translations
2. Verify which systems get certain points wrong
3. In many cases, we must try various (variants of) source sentences until we find one that exhibits just the error in question
4. Go through a process of examining the sentence and error(s) to confirm the "diagnosis"
5. Add it to the database
6. Annotate and post-edit the target

From the experience in QTLaunchPad and taking the learning curve into account, we estimate that it takes about 15mins for a segment (spread across multiple parties), to add one item to the test suite. The estimated costs for a substantial test suite for a new language pair are 5.000€-10.00€.

8 XML data format

Each XML data file contains a description of the elements and attributes at the beginning (thus eliminating the need for a separate file to document the semantics of the test suite file). A schema for validating test suite XML files is available at http://qt21.eu/deliverables/test-suite/test-suite-en-de.xsd. A short sample file, showing the internal documentation and some sample entries is presented below:

```
<?xml version="1.0" encoding="UTF-8"?>
<!--

   VERSION:           1.1
   GENERATION DATE:   2014-11-04
   HTML URL:          http://www.qt21.eu/deliverables/test-suite/test-suite-en-de.html
   XML URL:           http://www.qt21.eu/deliverables/test-suite/test-suite-en-de.xml

   ELEMENTS AND ATTRIBUTES:
     * <item> contains each test suite item
       - type indicates whether the item is from corpus data (corpus) or
         from the TSNLP grammar test suite (tsnlp)
       - xml:id provides the id value for the item
       - barrier-for provides which system types found the item to be a barrier
         (all, RbMT, SMT)
     * <source> contains the source segment
     * <target> contains the target segment with annotations
       - engine indicates which engine translated the segment (RbMT, SMT, human). If the
         engine is unknown, it is listed as –
         NOTE: human is included in a few instances because human translators frequently
                made errors which were then replicated by SMT systems based
               on bad human input
     * <issue> appears only within <target> or another <issue> element and indicates a
       specific issue in the target. If <issue> is nested it indicates that one issue is
       entirely enclosed within another. A typical example would be a punctuation error
       contained within a span marked for word order.
       - type contains the MQM issue type for the specific issue
     * <postedit> contains a post-edited version of the target
     * <notes> (optional) contains one or more <note> elements
     * <note> contains a single note about the item.
     * <em> indicates that its contents are to be emphasized, as with HTML <em>
-->
<items>
   <item type="corpus" xml:id="c001" barrier-for="RbMT">
      <source>"We want peace" said the president.</source>
      <target engine="SMT">
         <issue type="Quote mark type">"</issue> Wir wollen Frieden
         <issue type="Quote mark type">"</issue> , sagte der Präsident.
      </target>
      <postedit>„Wir wollen Frieden“, sagte der Präsident.</postedit>
   </item>
   <item type="corpus" xml:id="c002" barrier-for="all">
      <source> Cigarette plain packaging laws come into force in Australia </source>
      <target engine="HMT">
         <issue type="Mistranslation">Zigaretten-Ebenenverpackungsgesetze</issue>
          treten in Kraft in Australien ein
      </target>
      <postedit> Gesetze zur neutralen Verpackung von Zigaretten treten in Australien
         in Kraft.
      </postedit>
      <notes>
         <note>Complex compound term</note>
      </notes>
   </item>
   <item type="corpus" xml:id="c003" barrier-for="all">
      <source>Connection MOVIDRIVE® Master/Slave</source>
      <target engine="RbMT">
         <issue type="Word order">
            <issue type="Word form">Verbindungs</issue> -MOVIDRIVE® </issue> -Meister/
            <issue type="Terminology">Sklaven</issue>
      </target>
      <postedit>MOVIDRIVE®-Master/Slave-Verbindung</postedit>
      <notes>
         <note>Complex string of nouns</note>
      </notes>
   </item>
   <item type="corpus" xml:id="c004" barrier-for="all">
      <source>Drive vibrates</source>
      <target engine="RbMT">
         <issue type="Mistranslation">Fahrt</issue> vibriert </target>
      <postedit>Laufwerk vibriert</postedit>
   </item>
   <item type="corpus" xml:id="c005" barrier-for="all">
      <source>IN reduction</source>
      <target engine="SMT">
         <issue type="Terminology">In der</issue> Reduzierung </target>
      <postedit>IN-Reduzierung</postedit>
   </item>
   <item type="tsnlp" xml:id="g156" barrier-for="SMT">
      <source tsnlp-class="C_Complementation-Trivalent-Direct_object_VP_bare_infinitive">He
         has him leave.
      </source>
      <target engine="SMT"> Er <issue type="Mistranslation">hat</issue> ihn zu
         <issue type="Mistranslation">verlassen</issue> .
      </target>
      <postedit>Er veranlasst ihn, zu gehen.</postedit>
   </item>
<items>
```

The elements and attributes in the XML file map to the columns and items shown in the HTML as follows:

XML element/attribute	HTML output
`<item>`	Each `<item>` element corresponds to one row (test suite item) in the HTML output
`type`	Corresponds to the first letter of the **ID** value in the HTML: “corpus” (XML) = “c” (HTML); “tsnlp” (XML) = “g” (HTML)

XML element/attribute	**HTML output**
`xml:id`	**ID**
`barrier-for`	**Barrier for**
`<source>`	**Source**
`tsnlp-class`	**TSNLP class**
`<target>`	**Annotated target**
`engine`	**Trans. by**
`<issue>`	Individual annotated spans in the **Annotated target** column
`type`	MQM types listed in the **Issues** column
`<postedit>`	**Post-edited target**
`<notes>`	**Note(s)**
`<note>`	Individual item in the **Note(s)** column.
`<em>`	HTML `<em>`, used for emphasis/italics.

9 References

[1] Sabine Lehmann, Stephan Oepen, Sylvie Regnier-Prost, Klaus Netter, Veronika Lux, Judith Klein, Kirsten Falkedal, Frederik Fouvry, Dominique Estival, Eva Dauphin, Herve Compagnion, Judith Baur, Lorna Balkan, and Doug Arnold. TSNLP --- Test Suites for Natural Language Processing. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING-96)*, volume 2, pages 711-716, Center for Sprogteknologi, Copenhagen, 1996.

10 Changes from previous version

This version contains the following changes from version 2.0:

1. The XML format is described and documented
2. A description of the set of issue types used (with reference to D1.3.1) is provided, along with a link to the training materials used for annotation.
3. Usage scenarios are described together with ideas for ensuring sustainability
4. Effort estimation is provided
5. Various errata have been corrected.