TEI BY EXAMPLE



MODULE 8: CUSTOMISING TEI, ODD, ROMA

Ron Van den Branden

Edward Vanhoutte

Melissa Terras

Centre for Scholarly Editing and Document Studies (CTB) , Royal Academy of Dutch Language and Literature, Belgium, Gent, 9 July 2010

Last updated September 2020

Licensed under a Creative Commons Attribution ShareAlike 3.0 License

TABLE OF CONTENTS

4. Starting from a Minimal Schema....1
4.1 The Power of ODD....2
4.1.1 Generating a Schema....3
4.1.2 Generating Documentation for a Customisation....5
4.1.3 Generating an ODD File....7
4.2 What Does a Minimal TEI Customisation Tell Us?....9
4.3 Inspecting a Customisation in Roma....13

4. Starting from a Minimal Schema

If you point your browser to https://roma.tei-c.org/, following screen should appear:

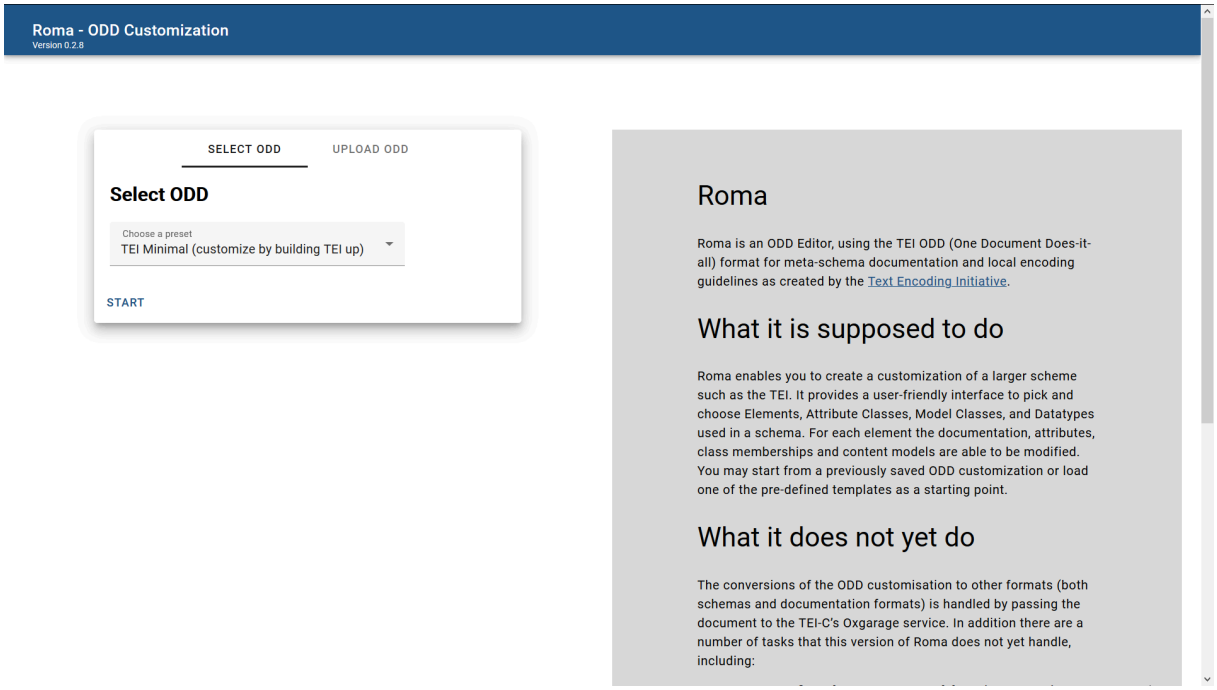


Figure 3. The start screen of Roma.

This is the start screen for new customisations, offering you a choice between two options:

- "Select ODD": start creating a customisation from one of the "official" TEI customisations
- "Upload ODD": start creating a customisation from an existing ODD file on your file system

For the purpose of this tutorial, we'll set out from a minimal customisation. Open the dropdown list and select the "TEI Minimal (customize by building TEI up)" template. This is a very simple customisation that only selects the TEI elements needed for producing valid TEI documents. Select this template and press the "Start" button.

This will produce a configuration screen for your fresh TEI customisation:

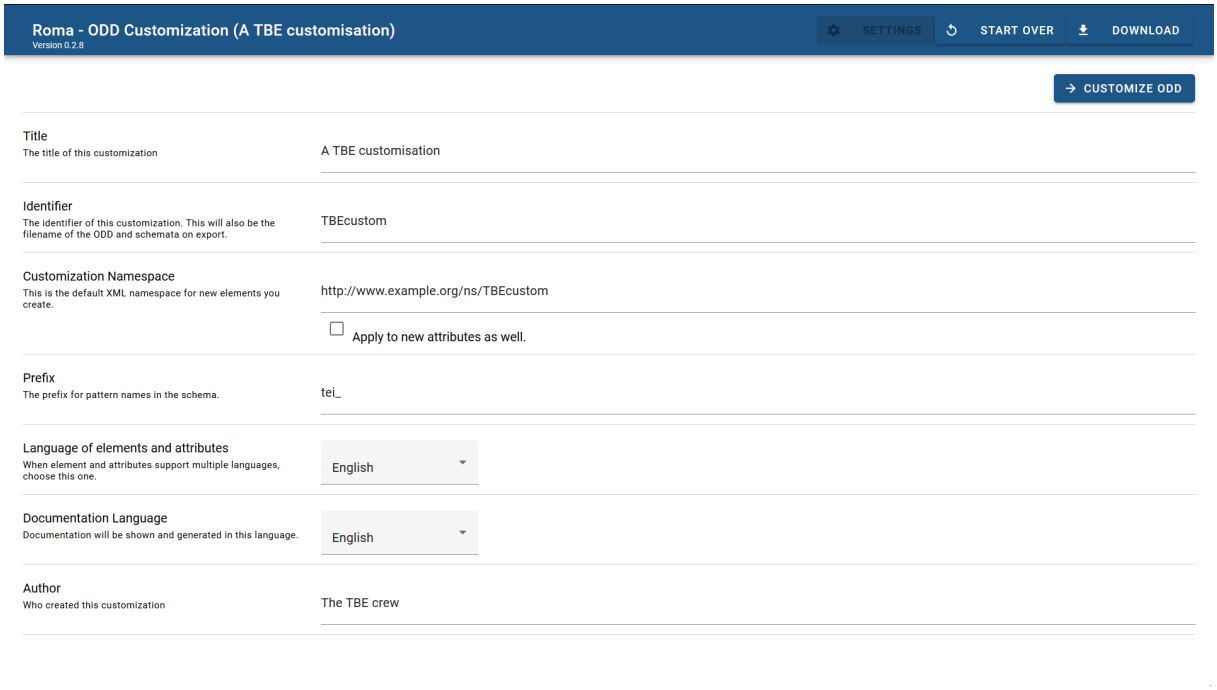


Figure 4. ODD configuration screen in Roma.

Here, you can enter details about your customisation file. Let's just personalise the metadata. Fill in:

- "A TBE Customisation" in the "Title" field;
- "TBEcustom" in the "Filename" field; and
- "The TBE crew" in the "Author" field.

That's it! We have created a first TEI customisation already, or at least an abstract representation in the Roma web interface. In order to use your TEI customisation in your project, you'll not only have to derive a schema to tell your XML parser what you consider a valid TEI document for your project, but most likely also human-readable documentation to tell your project team how they should encode the text phenomena they'll encounter in your project.

4.1 The Power of ODD

Remember how ODD stands for "One Document Does it all"? You can derive a number of different schema and documentation output formats from the single ODD file you're editing! The "Download" button at the top right of the Roma screen lets you choose between 3 types of output:

- ODD files

- schema files, in a number of flavours
- documentation files, in a number of output formats

Since these are frequent operations when customising TEI, they are treated in separate sections below.

4.1.1 Generating a Schema

Select the “Download” button at the top right in the Roma screen, and choose the schema language of your choice (the various schema language output formats are highlighted in yellow):

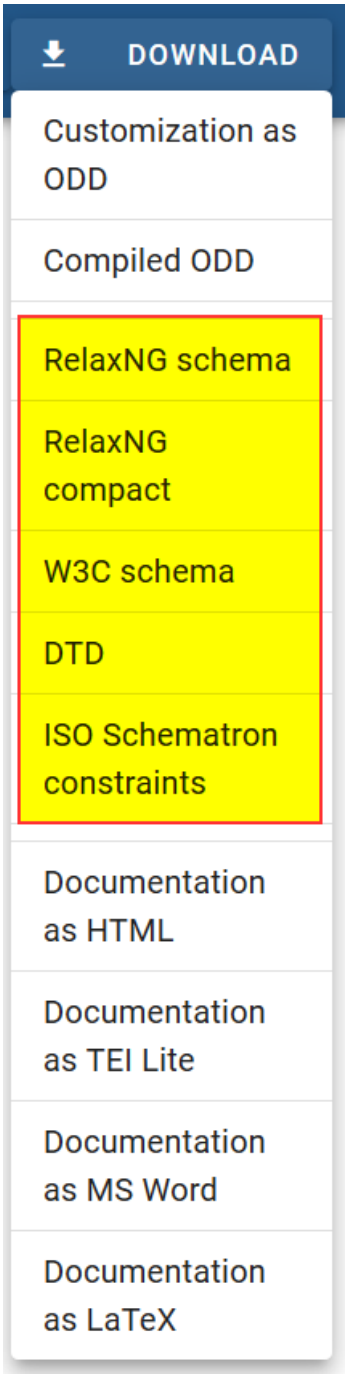


Figure 5. Schema download formats in Roma.

Roma can generate schemas in following schema languages:

- RelaxNG schema: a schema in the more verbose XML syntax of the RelaxNG language

- RelaxNG compact: a schema in the more concise compact syntax of the RelaxNG language
- W3C schema: a schema in the W3C XSD schema language
- DTD: a schema in the DTD language
- ISO Schematron constraints: a stand-alone file containing all ISO Schematron constraints that are present in the ODD file.

Although the schemas generated in these different schema languages are roughly equivalent, some schema languages offer more expressive power than others. The schema languages that offer most complete coverage of TEI features are RelaxNG (XML syntax) and W3C schema.

1

When choosing one of the schema flavours, your browser will download a file named **`TBEcustom`** (as we had specified earlier for our ODD in the "Settings" Roma tab). The file's extension depends on the schema format chosen: `.rng` (Relax NG XML), `.rnc` (Relax NG compact), `.xsd` (W3C schema), `.dtd` (DTD), or `.isosch` (ISO Schematron). You can store this file and use it to validate your TEI documents.

4.1.2 Generating Documentation for a Customisation

Select the "Download" button at the top right in the Roma screen, and choose the documentation format of your choice (the various documentation formats are highlighted in yellow):

.....

1 We'll not go into details about the differences between these XML schema languages here. For more information, Wikipedia is your friend: https://en.wikipedia.org/wiki/XML_schema.

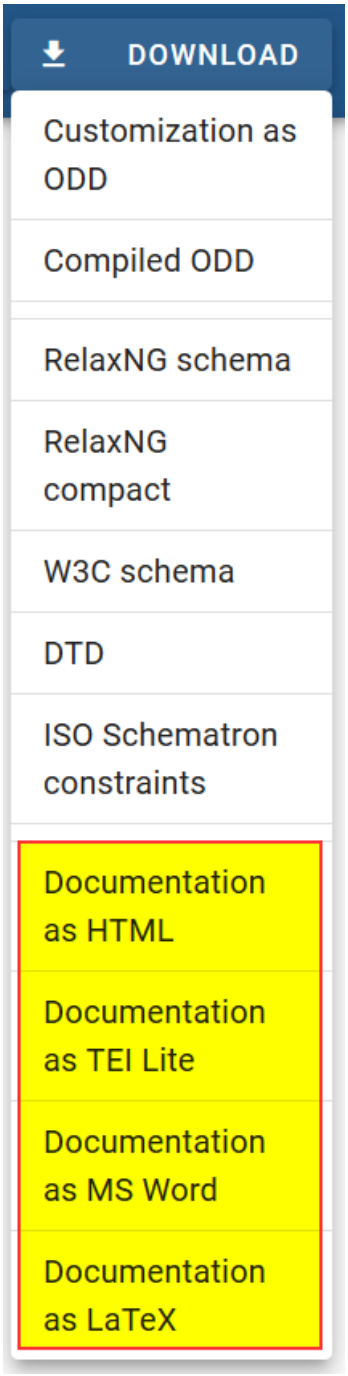


Figure 6. Documentation download formats in Roma.

Roma can generate documentation in following formats:

- HTML: a web page containing the documentation of the TEI customisation

- TEI Lite: a TEI Lite file containing the documentation of the TEI customisation
- MS Word: a Word file containing the documentation of the TEI customisation
- LaTeX: a LaTeX file containing the documentation of the TEI customisation

Make sure you save the file, and see how this produces a file named **`TBEcustom`**, either in HTML, TEI XML, DOCX, or LaTeX format. This documentation can serve as your project-specific TEI Guidelines, containing formal references for all elements in the schema, as well as any prose documentation present in the ODD file.

4.1.3 Generating an ODD File

We saved the best for last: without doubt, saving your customisation as an ODD file is *the* most important step of customising TEI with Roma. It will allow you (and others) to upload this customisation again for reuse, fine tune it further, and / or generate both schemas and documentation from this single source file again. Notice that, although all changes you make in the Roma web interface are being recorded automatically, you still have to save your customisation as an ODD file. In order to do so, all you have to do is hit the "Download" button at the top right in the Roma screen, and choose one of the ODD formats (the ODD formats are highlighted in yellow):

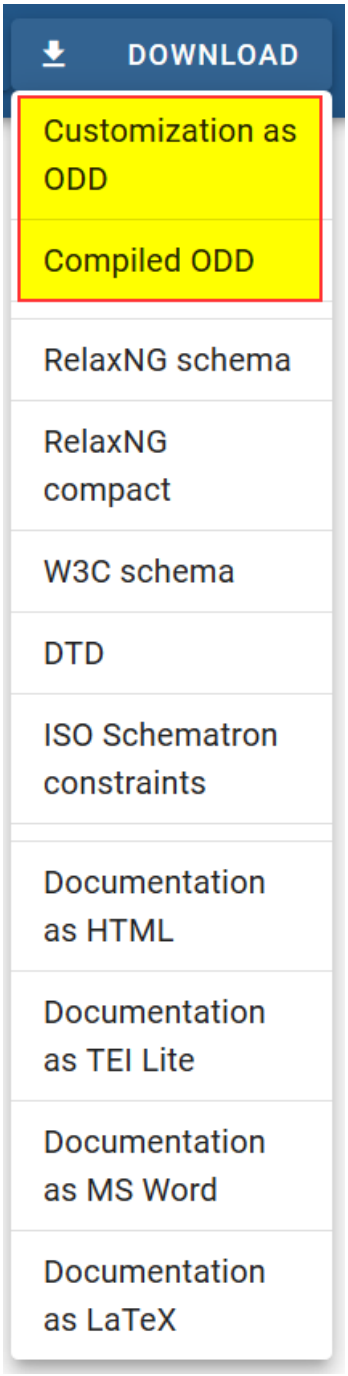


Figure 7. ODD download formats in Roma.

Roma offers following ODD formats:

- Customization as ODD: an ODD file, referencing the various TEI components

- Compiled ODD: an expanded ODD file, containing full documentation and with all references to the TEI components resolved

You'll probably need the first option: "Customization as ODD." This will create a file called **`TBEcustom.odd`**, which is a TEI XML file you can open in any plain text or XML editor. Notice, once again, how the file name corresponds to the one specified for our customisation in the "Settings" Roma tab.

SUMMARY

Roma provides a visual interface to create TEI customisations, either from any of the customisation templates provided by the TEI Consortium, or from a previously saved ODD file on your file system. Customisations can be edited in Roma and exported as an ODD (One Document Does it all) file, from which both a TEI schema and accompanying documentation can be derived in a number of output formats. The ODD file is the core of your TEI customisation.

4.2 What Does a Minimal TEI Customisation Tell Us?

Before proceeding, let's have a closer look at what we've done. As mentioned before, the **`TBEcustom.odd`** file we had stored earlier, is just a TEI XML file. Let's have a look what's inside!

```
<TEI xmlns="http://www.tei-c.org/ns/1.0" xmlns:rng="http://relaxng.org/ns/
structure/1.0" xml:lang="en">
  <teiHeader>
    <fileDesc>
      <titleStmt>
        <title>A TBE Customisation</title>
        <author>The TBE crew</author>
      </titleStmt>
      <publicationStmt>
        <publisher>TEI Consortium</publisher>
        <availability status="free">
          <licence target="http://creativecommons.org/licenses/by-sa/3.0/">
           Distributed under a Creative Commons Attribution-ShareAlike 3.0 Unported
           License </licence>
          <!-- ... -->
```

```
      </availability>
    </publicationStmt>
    <sourceDesc>
      <p>Created from scratch by James Cummings, but looking at previous tei_minimal
       and tei_bare exemplars by SPQR and LR.</p>
    </sourceDesc>
  </fileDesc>
</teiHeader>
<text>
  <body>
    <head>A Minimal TEI Customization</head>
    <p>This TEI ODD defines a TEI customization that is as minimal as possible
     and the schema generated from it will validate a document that is
     minimally valid against the TEI scheme. It includes only the ten required
     elements: <list rend="numbered">
      <item><gi>teiHeader</gi> from the header module to store required
       metadata</item>
      <item><gi>fileDesc</gi> from the header module to record information about this
       file</item>
      <item><gi>titleStmt</gi> from the header module to record information about the
       title</item>
      <item><gi>publicationStmt</gi> from the header module to detail how it is
       published</item>
      <item><gi>sourceDesc</gi> from the header module to record where it is
       from</item>
      <item><gi>p</gi> from the core module for use in the header and the body</item>
      <item><gi>title</gi> from the core module for use in the titleStmt</item>
      <item><gi>TEI</gi> from the textstructure module because what is a TEI file
       without that?</item>
      <item><gi>text</gi> from the textstructure module to hold some text</item>
      <item><gi>body</gi> from the textstructure module as a place to put that
       text</item>
    </list></p>
    <schemaSpec ident="TBEcustom" start="T
    EI" prefix="tei_" targetLang="en" docLang="en">
      <!-- required minimal header elements -->
      <moduleRef key="header" include="teiHeader fileDesc titleStmt publicationStmt
       sourceDesc"/>
```

```
        <!-- required core elements (p and title for use in titleStmt) -->
        <moduleRef key="core" include="p title"/>
        <!-- required textstructure elements (TEI, text, and body) -->
        <moduleRef key="textstructure" include="TEI text body"/>
        <!-- required module tei instantiates lots of classes used for further expansion
         of this odd -->
        <moduleRef key="tei"/>
      </schemaSpec>
    </body>
  </text>
</TEI>
```

Example 1. A minimal TEI customisation ODD (download).

One immediate observation is that an ODD file is indeed just a regular TEI document, with a <TEI> root element, containing a <teiHeader> element and a <text> element. Remember the metadata you entered in the "Settings" tab in Roma, and see how it is reflected at the proper places inside the <teiHeader>. However, the most interesting bits are inside the <body> part.

Apart from regular body content, as illustrated by the <p> contents of our minimal TEI customisation, an ODD file contains a specific <schemaSpec> element. This element indicates a formal definition of a TEI schema. It has a mandatory @ident attribute, supplying an identification code for the schema. The language of the documentation can be specified with an optional @docLang attribute; when necessary a @targetLang attribute can specify what language to use for element and attribute names. The @prefix attribute specifies the prefix that will be added to the names of TEI patterns in the customisation. The @start attribute identifies the root element(s) of the customisation: in this case, it will produce a schema that only allows the <TEI> element as root element for valid TEI documents.

3

.....

3 Since an ODD file is just a regular TEI file, the formal schema specification in <schemaSpec> can be completed with a detailed prose documentation (and in fact, the concept of an ODD file is *intended* to contain both formal specifications and prose documentation), using the regular TEI elements in the <body> part. Our current example only contains very minimalistic prose documentation; for an excellent example, see the documentation in the TEI Lite ODD file, and compare it with the different documentation files offered at https://tei-c.org/guidelines/customization/Lite/.

The <schemaSpec> element is the core of any ODD file, specifying the formal definition of a TEI schema. A TEI customisation can be constructed by referring to definitions of existing TEI objects (element and attribute classes, datatypes, and macros), or—as will be covered later in this tutorial—by declaring new objects as well. In our example so far, the schema specification only contains references to four of the predefined TEI modules. Each module is being referred to in a separate <moduleRef> element, whose @key attribute is pointing to the formal identifier of that module in the TEI source.

When a module is selected with <moduleRef> in an ODD file, by default all elements and attributes of that module are incorporated in the customisation. In our example, this is the case with the **tei** module. As we have seen, this is a special module, which doesn't define any elements and attributes itself, but instead a lot of classes, datatypes, and macros that are being used in all other TEI modules. Three of those other modules are being referenced in our current ODD file. Yet, only a very small number of elements they define are being included in our customisation, because the <moduleRef> elements have an @include attribute, where only those elements that should be included are enumerated as a white-space separated list. If you count them, our customisation only selects the 10 elements that are required for minimally valid TEI documents.

Notice, how the @include mechanism has a counterpart: it is equally possible to specify which elements *not* to include in a module by enumerating them in an @except attribute. This will only exclude those elements, but will include all others. Both mechanisms offer convenient shortcuts to either include or exclude large numbers of elements. It's important to remember that you can only use either the @include or the @except attribute on a <moduleRef> element, but not both.

SUMMARY

TEI groups its different elements and their attributes in 21 modules. When defining a TEI customisation, these modules can be referred to in an ODD file. An ODD file is just a regular TEI document with a specific element for defining a TEI schema: <schemaSpec>. An identification for the schema must be provided in an @ident attribute. Inside the schema specification, modules can be referenced with a <moduleRef> element, naming the module in a @key attribute. When a TEI module is included in a customisation, by default all of its elements and attributes

will be included. In order to include specific elements only, the names of these elements can be enumerated in an @include attribute. Likewise, if only some of the elements of a module should be excluded, they can be enumerated in an @except attribute on <moduleRef>.

4.3 Inspecting a Customisation in Roma

As we have introduced in section 2, the TEI is a highly sophisticated library, holding model and attribute classes, macros, and datatypes. Customising TEI requires some knowledge of how they are organised. Full documentation can be found in the TEI Guidelines, but fortunately, Roma is conceived as a convenient online store for selecting the exact TEI components you need, and doing so, it shows you around in the internals of TEI as well. Let's start editing our current customisation where we have left it by hitting the "Customize ODD" button at the top right in the Roma screen:

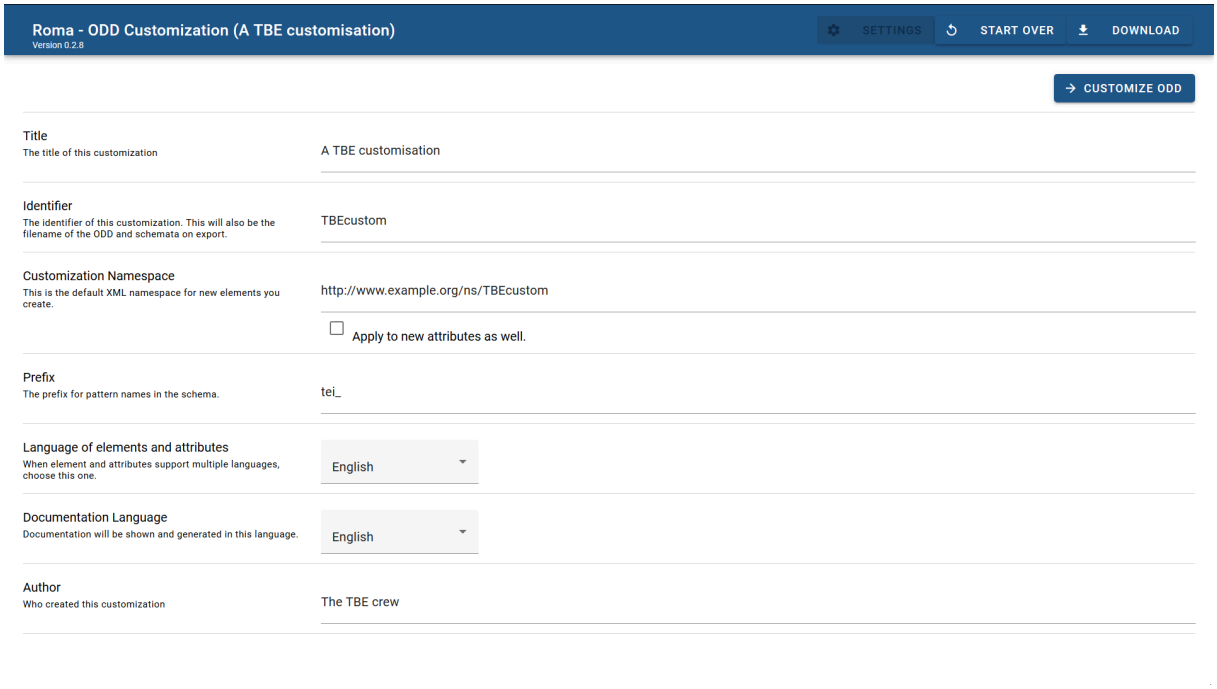


Figure 8. Start customising an ODD file with the "Customize ODD" button.

This will take you to the main Roma dashboard. You can tell you're editing the right customisation from the title at the top of the screen, which reads "Roma - ODD Customization (A TBE customisation)."

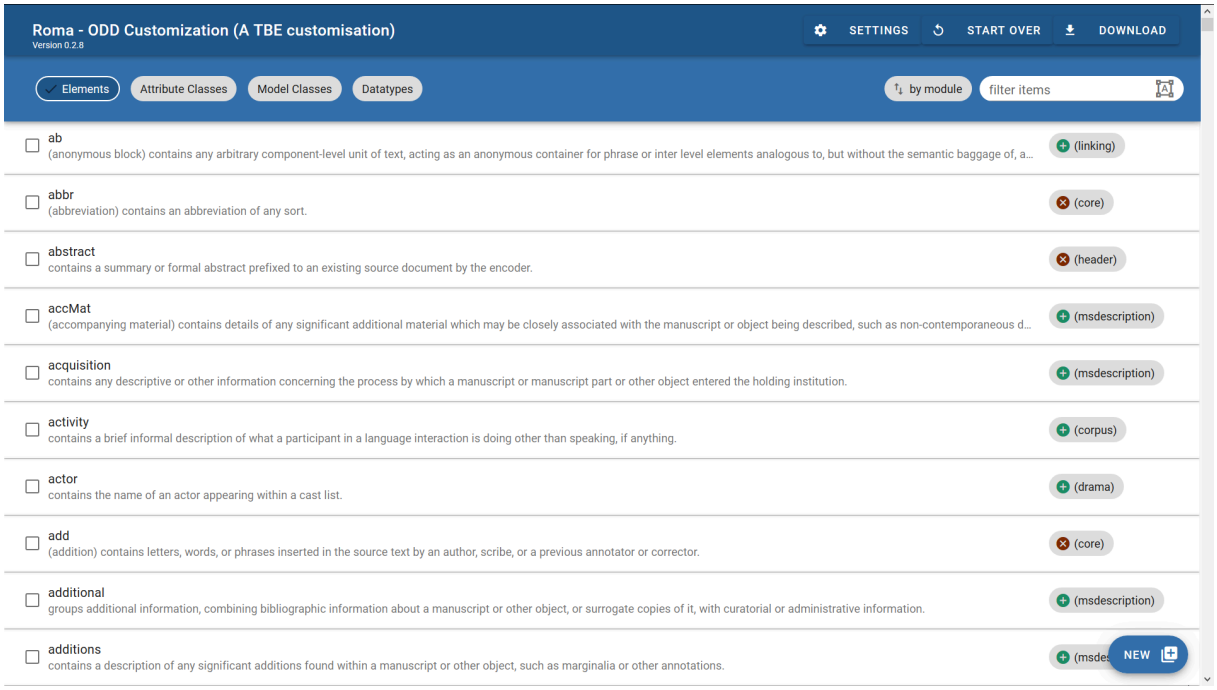


Figure 9. Roma main screen.

Below the title, you see 4 different tabs at the top of the screen:

- "Elements": an overview of all available TEI elements, allowing you to select and customise those you need
- "Attribute Classes": an overview of all available TEI attribute classes, allowing you to select and customise those you need
- "Model Classes": an overview of all available TEI model class definitions, allowing you to select and customise those you need
- "Datatypes": an overview of all available TEI datatype definitions, allowing you to select and customise those you need

Let's check how the <moduleRef> elements we've seen in our current TEI customisation are being reflected in the Roma interface. When you select the "Elements" tab, you'll notice that only a very small number of elements are ticked. Indeed, from the long list of available TEI elements, only 10 have been selected: <body>, <fileDesc>, <p>, <publicationStmt>, <sourceDesc>, <TEI>, <teiHeader>, <text>, <title>, and <titleStmt>. Notice, how the default ordering in Roma is alphabetical; you can also click the "by module" button at the top right of the Roma screen. This will order the elements (or attribute classes, or model classes, or datatypes) alphabetically: first per module, and next by name. This sorting "by module" gives you an idea of what TEI components are

being defined in each of the modules. This is broader than elements: if you look at the other tabs, you'll notice that a lot of attribute classes, model classes, and datatypes are being included automatically when a TEI module is included via <moduleRef> in a TEI customisation.