



TSL editor user manual

1. Introduction.....	3
1.1. Requirements and installation.....	3
1.2. Folder contents.....	4
2. TSLEdit application: general information.....	4
2.1. Logical structure.....	4
2.2. Main window.....	5
2.3. Main window menus.....	5
2.3.1 Modify menu.....	5
2.3.2 Tools menu.....	6
2.4. Fields management	6
2.4.1 Single language and multi-language field values add/delete.....	7
2.4.2 Multi-language address field values add/delete.....	7
2.4.2.1 Multilanguage attribute example.....	8
2.4.3 Field that provide default values to be inserted add/delete.....	9
2.4.4 Qualification Element add/delete.....	9
2.5 Mandatory fields.....	10
3 TSLTool application: usage.....	10
3.4 Creating a new Trust Service Status List	10
3.5 Scheme information section.....	10
3.6 TSP information section.....	13
3.7 Service information section.....	14
3.8 Service history information section.....	16
3.9 Qualification Element section.....	16
4 TSLEExport.....	18
5 Signature and validation.....	20
5.4 Signature and validation features.....	20
5.5 Signing a file step-by-step.....	20
5.5.3 PKCS12.....	20
5.6 Verifying a file.....	20
5.7 Validate a file.....	21
6 Technical notes.....	21

TSL editing tool

1. Introduction

The software here presented has been designed to make easier all the editing activities related to a TSL software object. It produces as output an XML file fully compliant with the latest version of the ETSI -TS 102.231 v3.1.1 (2009/06) standard document [1] plus the amendments and corrections stated in the Technical specifications for a Common Template [2], explained also in the Annex L of [1].

In this document an overview of the functionalities available and a short explanation of the meaning for each of the fields present in the different masks is given. For further specific details and full explanations the user can refer to the official documents referred in [1] and the model developed in the framework of the WESIGN project ([3] and [4]).

The application is based on a graphical interface that allows a quick building of a TSL file object and a simple access to the signature features.

The main features of the TSLEdit tool application can be summarized as follows:

- (Create a new) / (Open and parse an existing) ETSI TS102231v3.1.1-compliant TSL file object
- Review the XML structure of the (created) / (under development) TSL file object and save it in an XML file
- Validate the XML document produced against the predefined xsd schema
- Present the saved TSL in a human-readable format through an external window, giving to the user the possibility to print it
- Export the TSL in pdf format, expanding the certificate attributes of the services listed as stated in [1]
- Sign the produced TSL file with an XML signature through a certificate on file (PKCS12)
- Check the validity of the signature on a signed TSL file, also against a supplied “trusted” signer certificate

1.1.Requirements and installation

The TSLEdit tool application has been designed to work on the following platforms:

- MS Windows 2000
- MS Windows XP
- MS Windows 2003
- MS Windows Vista

The TSLEdit tool needs the .NET framework 2.0 correctly installed on the local machine to perform several operation (signature creation and verification, xml validation).

The MS Arial Unicode fonts should be present on the local machine to correctly visualize the Multilanguage fields in non-latin languages.

1.2.Folder contents

The application doesn't need any installation procedure. To start the tool is enough to run the TSLEdit.exe application file.

The application folder contains the following objects:

TSLEdit.exe	the executable application
XMLEnvSig.exe	a console application called by TSLEdit.exe during the signing and verification operations. It is not intended for a standalone use.
TSLEExport.exe	a graphical application that present to the user the TSL in a human-readable format, offering the possibility export it in a pdf file.
Certs folder	contains the certificate files used for the testing activities
Doc folder	contains a copy of the present manual
Schema folder	contains the ETSI TS 102231 v2.1.1 schema referenced by ETSI with the standard document [1] plus the amendments stated in [2]
SignedOutput folder	contains a sample TSL, with and without the signature, developed with the received contributions of the other partners of the WESIGN project

For more detailed information about signing and verifying a TSL file see the appendix.

2.TSLEdit application: general information

2.1.Logical structure

The editing of a TSL is divided in four different pages, each page related to a specific section as defined in [1]. The section are logical subsection in which a TSL is divided. They are four, outlined below in a tree-style view:

- Scheme Information section
 - ◆ Trust Service Provider(s) Information section
 - Service(s) information section
 - Qualification Element(s)
 - Service History(ies) information section
 - Qualification Element(s)

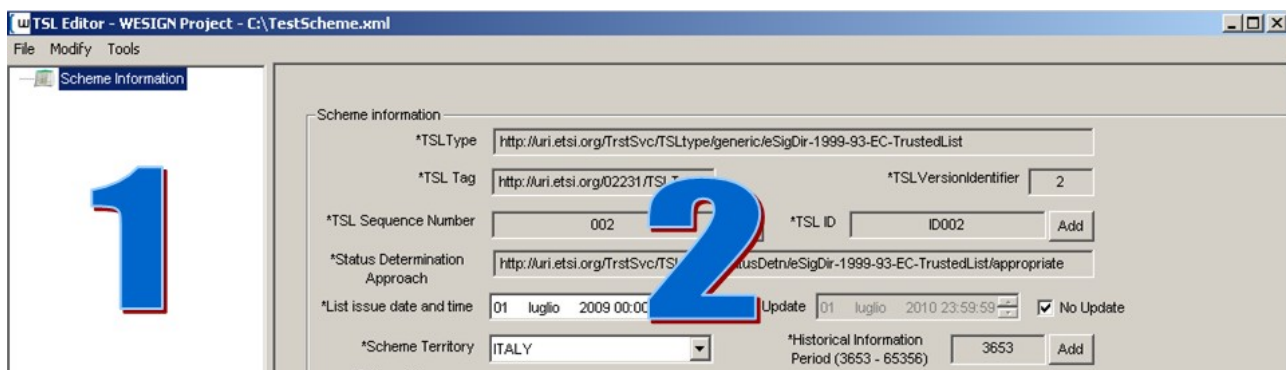
The term “section” roughly indicates a set of attribute that can be grouped together to describe one of the entity involved in a TSL object.

Each of the five pages available in the TSLEdit application presents the edit boxes corresponding to the specific section where the values of the attributes describing the entity involved has to be entered.

2.2.Main window

The active area of the main window of the TSLEdit application is divided in two different areas:

- **Area 1:** The tree visualization “area”: contains the graphical tree representation of the current TSL, opened from a previously saved file or under construction.
- **Area 2:** The main editing “area”: contains the “read-only” edit boxes related to the specific section under editing.
Inside this “area”, on the right side of each of the text-box that requires to be compiled, there is a pair of Add/Delete buttons, that allow to enter or delete values for the corresponding field. Moreover, drop-down-list controls are present for multi-language or multi-values attributes, allowing to select which node value has to be displayed.



See below for more details.

2.3.Main window menus

On the top left corner there are three system menu. Besides the File menu, that contains the usual Open, Close, Save functions, there are two more menu that will be seen in more details.

2.3.1Modify menu

The Modify menu contains the function that allow to add an entity to the current tree node. The menu entries will be enabled or disabled depending on which kind of node is currently highlighted in the tree, the area previously indicated as “Area 1” (i.e. which mask is currently shown to the user).



All the editing functionalities that can be accessed from this menu are also available right-clicking on the tree node to which the new element (TSP, service or service history instance) has to be appended

2.3.2 Tools menu

The tools menu allows the access to the following feature.



XMLSource: opens a popup window in which can be reviewed the actual XML structure of the document under development. Please remember that the fields that provide values through a drop-down list could not have been saved inside the XML document. To ensure that all the values inside the current mask have been inserted press the “Apply” button or, alternatively, switch to another node to cause the application writing into the file.

Verifiy Signature: performs a validation of the signature node contained in the XML document. Details are given in chapter 5.

Export signature certificate:

allows the user to save the Scheme Operator’s certificate with which the TSL has been signed. If the signature node could not be found a message box will alert the user.

Validate:

performs the validation of the XML syntax over the currently opened TSL against the predefined schema contained inside the application folder. If the xsd schema could not be found a message box will alert the user.

Export human readable version of this TSL:

performs the export of the TSL created in a human readable format, opening an external viewer. More details are given in chapter 4.

2.4. Fields management

All the language attribute drop-down lists are initialized with the default language “EN”. **Please remember that the English text-value is mandatory for all the multi-language field** to produce a valid TSL conformant to [2].

On the bottom of each section’s editing masks there is an “Apply” button. Press it after editing the fields to force the writing of all the fields inserted and the update of the tree structure and images.

2.4.1 Single language and multi-language field values add/delete

Pressing the “Add” button (where available) a pop-up modal window (floating, not inserted inside the main window) will appear offering a mask to enter the value for the selected field.

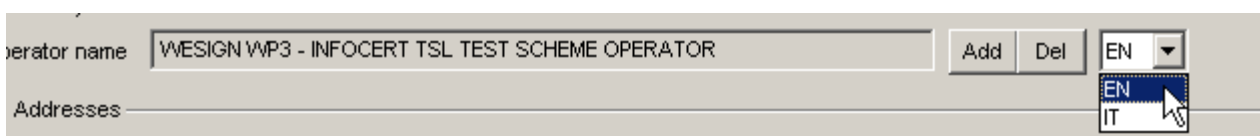


The mask showed will allows the user to type in the field value and, if needed, to select the language attribute that has to be inserted into the corresponding XML node inside the TSL document. If the language attribute is not required for the selected field, the language drop-down control will be disabled.

Changing the language attribute will cause the application to check into the XML document if any content for the selected node is already available for the language attribute newly selected. If is present, the corresponding field is updated with the retrieved node content. Vice versa, the field is blanked.

To add a new value, always select first the right language attribute on the drop-down list before entering the field content, or it will be erased when switching to another language.

After pressing the “Confirm” button the user will be brought back to the main window: here the value inserted is added to the XML document and inside the corresponding text-box. If also a language attribute pertains to the field, it is added to the list present on the left side of the text-box.



2.4.2 Multi-language address field values add/delete

The pop-up window will appear after pressing the “add” button for an address field. The usage of the language control is the same as described above.

As described above, changing the language attribute will cause the application to check in the XML document if the content for the selected node is available also for the language attribute newly selected.

After pressing the “Confirm” button the user will be brought back to the main window: here the value inserted is copied inside the XML document along with the language attribute and inside the corresponding text-box and drop-down list.

2.4.2.1 Multilanguage attribute example

To make a clarifying example, taking the “SchemeOperatorName” field of the Scheme Information mask:

- Press the “Add” button on the left side of the “SchemeOperatorName” text-box
- Inside the pop-up window that will appear, preselect the preferred language for the field value that is to be entered, for example “EN”.
- Fill with the value “Schema Operator Certificazioni Digitali xyz” the editable text-box
- Press the “Confirm” button

The value just inserted is copied into the text-box in the main window and the language selected is added to the language list on the right side of the text-box.

- To enter another value, press again the “Add” button on the left side of the “SchemeOperatorName” text-box
- Inside the pop-up window that will appear, preselect again another language attribute, say “IT”.
- Fill with the value “Scheme Digital Certification xyz” the editable text-box
- Press “Confirm” again.

The value just inserted substitutes the preceding one inside the text-box in the main window and the language selected, if not already present, is added to the language list on the right side of the text-box and selected as the current.

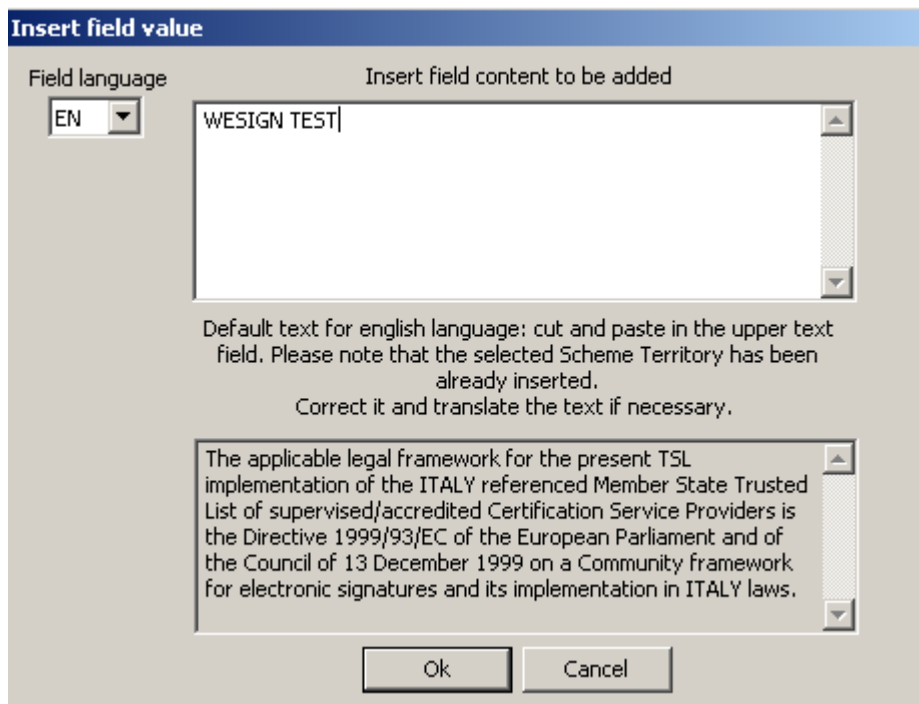
Below is an excerpt from the resulting XML file:

```
....
<tsl: SchemeOperatorName >
  <tsl:Name xml:lang="EN">Scheme Operator Digital Certification xyz</tsl:Name>
  <tsl:Name xml:lang="IT">Schema Operator Certificazioni Digitali xyz</tsl:Name>
</tsl: SchemeOperatorName >.....
```


2.4.3 Field that provide default values to be inserted add/delete

For several fields belonging to the scheme information section, namely the “TSL Legal Notice”, “Scheme Information URI” and “Scheme type Community Rules” fields, the annex referred in [2] provide a default English text to be inserted as a mandatory content, to which the user can eventually add more content..

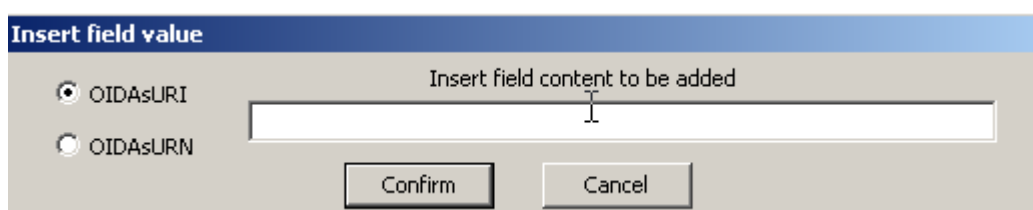
Pressing the “Add” button, the pop-up window that will appear will offer on the upper side a multi-language editable text-box and on the lower side a read-only text-box carrying the indication of the default text to be inserted.



Please notice that the tool will automatically substitute the placeholder contained in the original text *[name of the relevant member state]* with the currently selected country name showed inside the “Scheme Territory” control.

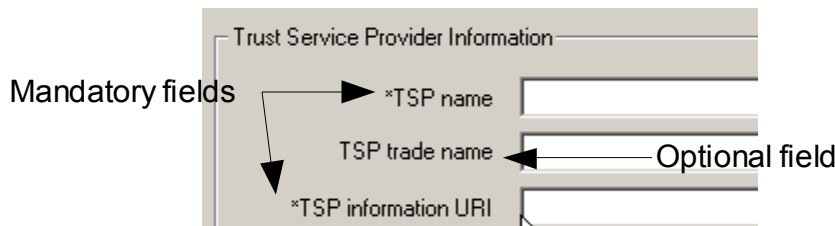
2.4.4 Qualification Element add/delete

The Qualification Element is added through a pop-up window similar from the preceding ones. Inside it the user can use a single editable text-box and, if needed, perform an attribute selection for the XAdES:Identifier Qualifier attribute, as shown below.



2.5 Mandatory fields

Not all the fields that pertain to a mask are mandatory. The minimum subset of fields that must be filled to produce a valid TSL object have the name label tagged with an asterisk.



If the user try to save a TSL to a file leaving out some of those fields empty, a dialog window will pop up, listing the missing values and asking for a confirmation to proceed with the saving anyway.

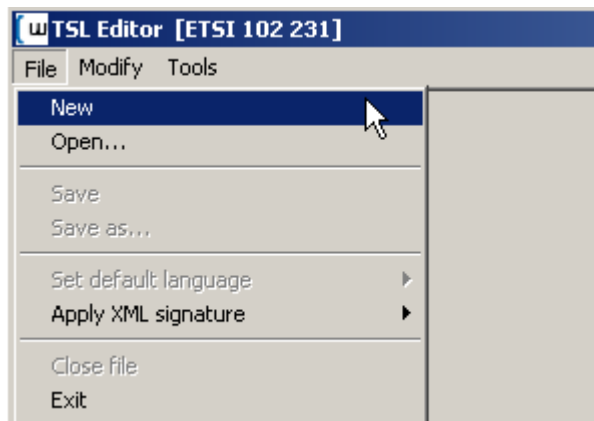
The non-mandatory fields, if empty, will be removed from the final xml document structure.

3 TSLTool application: usage

3.4 Creating a new Trust Service Status List

Starting the TSLEditexe, an empty dialog window will show up. On the top right corner there are some sytem menu to call the main functions

To start the creation of a new TSL file, select "New" from the "File" menu.



3.5 Scheme information section

The first mask shown is the one relative to Scheme Information section. Several fields are already filled by default with mandatory values fixed by the standard.

The scheme section defines all the attribute of the entity that publish and signs the TSL object, fixing its content.

Scheme information		
*TSLType	http://uri.etsi.org/TrstSvc/TSLType/generic/eSigDir-1999-93-EC-TrustedList	
*TSL Tag	http://uri.etsi.org/02231/TSLTag	*TSLVersionIdentifier 3
*TSL Sequence Number	<input type="text"/> Add	*TSL ID <input type="text"/> Add
*Status Determination Approach	http://uri.etsi.org/TrstSvc/TSLType/StatusDetn/eSigDir-1999-93-EC-TrustedList/appropriate	
*List issue date and time	21 luglio 2009 10:13:54	*Next Update 21 luglio 2009 10:13:54 <input type="checkbox"/> No Update
*Scheme Territory	AUSTRIA	*Historical Information Period (3653 - 65356) <input type="text"/> Add
*Scheme Name (scheme territory prefix will be added)	<input type="text"/> Add Del	EN
*Scheme Operator name	<input type="text"/> Add Del	EN
Physical Addresses		
*Street address	<input type="text"/> Add Del	EN
*Locality	<input type="text"/>	*Postal Code <input type="text"/>
StateOrProvince	<input type="text"/>	*Country Name AUSTRIA
*Scheme Operator email address	<input type="text"/> Add	
*TSL Legal Notice	<input type="text"/> Add Del	EN
*Scheme Information URI	<input type="text"/> Add Del	EN
*Scheme type community rules	<input type="text"/> Add Del	
Distribution Points	<input type="text"/> Add Del	
Pointers to other TSLs	<p>*****</p> <p>Other TSL pointer</p> <p>Note: While waiting for the ETSI TS 102 231 compliant implementation of the EC compiled list of links towards Members State's TSL implementation of their TSLs, these fields SHALL NOT be used.</p> <p>*****</p> <p>Add Del</p>	
Additional Information	<p>Add Del</p> <p>EN</p>	
Extensions:	Extension node name <input type="text"/> Extension content <input type="text"/> Add Del	Extension nodes list <input type="text"/>
Critical = true <input type="checkbox"/>		
Apply		

Scheme Name: the name of the current scheme. In the context of the WESIGN project it will be a fixed value containing an Id of the project itself. As required by [2], to the SchemName the 2-letter country code prefix will be prepended.

Scheme Information URI: the field is constituted by a mandatory text as stated in [2] plus an optional URI where the scheme-specific information on the current schema can be found.

Status determination approach: the type of supervision by the scheme operator. The value is defined in [2]

List issue date and time: the publishing date of the current TSL

Next update: The foreseen next publishing date of this TSL. This attribute can be NULL, meaning that the scheme operator does not have scheduled any further publication of this TSL.

TSL sequence number:	a progressive number that identifies the chronological series of the TSLs published by the scheme operator.
TSL ID:	a unique identifier of this TSL. It will be referred as URI in the XML signature node. The Id fields has to start with a letter and must not contain colon character (:)
Scheme territory:	the country code in which the scheme is established
Historical information period:	indicates the numbers of days during which the TSL information will be retained by the scheme operator. It can vary from 3653 (10 years) to 65534 (179 years (!), as defined in [1] and [2]). A value of 65535 means indefinite duration.
Scheme type/ community rules:	<p>the field is constituted by a mandatory text plus optionally one or more URIs that can be inserted in order to identify specific policies/rules. Those URIs that can be read with one of the following meanings:</p> <ul style="list-style-type: none">the services listed have been assessed against the policy/rules referred by the URI listed (Optional).orthe community of users that use TSL has to agree/can be identified by the policy/rules referred by the URI listed (Optional).
TSL legal notice:	the field is constituted by a mandatory text as stated in [2]
Scheme information extensions:	an extension free text field to add information to the specific trust service provider (Optional)
Other Tsl pointers:	currently this fields is not implemented, as stated in [2]
Additional information:	currently this fields is not implemented, as stated in [2]
Scheme operator name:	the name of the scheme issuer
Email address:	a valid email address of the Scheme Operator (single value)
Scheme operator address:	
Street address:	the postal street address of the Scheme Operator.
Locality:	the town/city in which the provider is legally established
Postal Code:	alphanumeric postal code
State or province	a regional identifier, where applicable (Optional).
Country name:	the country in which the Scheme Operator is established. The value will be translated before the insertion in the final XML document in it's two-letter ISO 3166-1 country code
Distribution Points:	URI of the distribution points of this TSL

Scheme Extensions:

a free fields through which the Scheme Operator can specify every additional information about the TSL. It is possible to choose both the node name and the node content.

3.6 TSP information section

The TSP information section accomodate the details to fully describe a trust service provider. It's basically divided in two distinct sections, one relative to the TSP name and information URIs and one relative to the physical and the electronic address of the provider.

Trust Service Provider Information

*TSP name Add Del EN ▼

TSP trade name Add Del EN ▼

Physical Address

*Street address Add Del EN ▼

*Locality *Postal code

StateOrProvince *Country

*TSP email address Add

*TSP information URI Add Del EN ▼

Trust Service Provider Extensions

Extensions: Extension node name Extension content Extension nodes list ▼

Critical = true ☐

Add Del

Apply

TSP Name:

the name of the trust service provider

TSP Trade name:

the “commercial” name of the service provider (Optional).

TSP Information URI:

an URI supplied by the trust service provider through which the users can access specific TSP related information

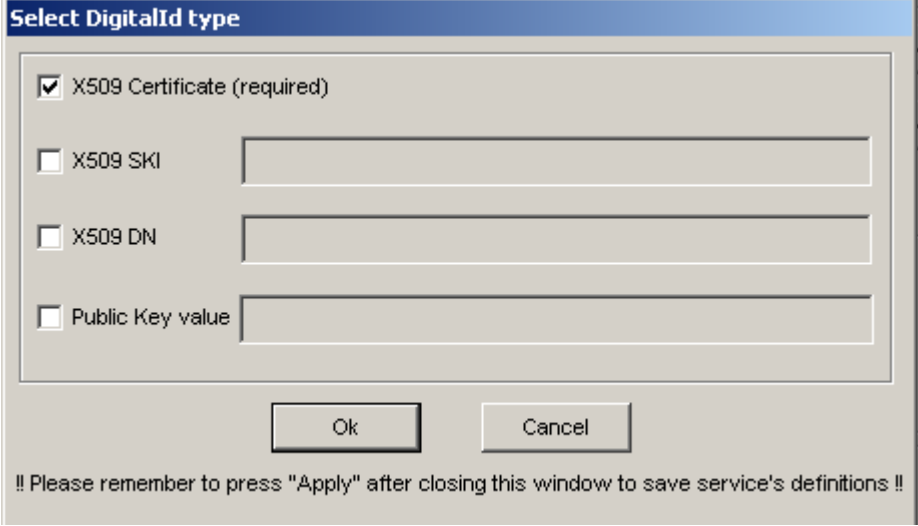
TSP Information Extension:

an extension free text field to add information to the specific trust service provider. It is possible to choose both the node name and the node content. (Optional).

The Trust Service Provider Address' fields are analogous to the ones formerly described for the Scheme operator address. See the descriptions above.

3.7 Service information section

The service section accommodate the details of a single service. When a new service is added, the user will be asked to select the kind of Digital Identity attribute that will identify the service through the dialog illustrated below.



The dialog box is titled "Select DigitalId type" and has a blue header bar. It contains four options, each with a checkbox and a text input field to its right:

- ☒ X509 Certificate (required)
- ☐ X509 SKI
- ☐ X509 DN
- ☐ Public Key value

At the bottom of the dialog are two buttons: "Ok" and "Cancel". Below the buttons is a message: "!! Please remember to press 'Apply' after closing this window to save service's definitions !!".

It's mandatory to use an X509 certificate to identify the specific service. Supplying the certificate, the SKI (Subject Key Identifier) attribute, if present, and the DN (Distinguished Name) string value will be fetched directly from the certificate.

After the selection of the Digital Identity attribute, the service section mask will be presented.

Service Information

*Service type identifier:

*Service name:

*Service current status:

*Current status starting time:

SchemesServiceDefinitionURI:

TSPServiceDefinitionURI:

Service Supply Points

Supply point URI:

Service Information Extensions

Extensions:	Extension node name	Extension content	Extension nodes list
	<input type="text"/>	<input type="text"/>	<input type="button" value="Add"/> <input type="button" value="Del"/> <input type="button" value="Extension nodes list"/>

Critical = true ☐

Service Additional Information

Additional information URI:

Additional information string:

Additional Information node present: FALSE

Service Extensions

Expired Certificate Revocation Info: Expired certificate info present: FALSE

Service Qualification Element

Service Information Extension present: FALSE

- Service Type Identifier:** the type of service delivered. The selection is made through a drop down list.
- Service name:** the name of the delivered service. If available, it will be filled with the DN value extracted from the certificate
- Service current status:** the actual status of the service, among the choices available in the list. Previous status/events in the provider history can be inserted in the following service history section
- Status starting time:** the starting time of the actual status.
- Scheme service definition URI:** an URI supplied by scheme operator in which the kind of service is described (Optional)
- Service supply points:** an URI where the users can access the service (Optional)
- TSP service definition URI:** an URI supplied by the service provider where the kind of service is described (Optional)
- Service information extension:** The field is composed of different kinds of fields, also in relation with the type of service specified.

Extensions: like for the Scheme and TSP extensions, this is a free text fields available for every kind of additional information about the service could be needed to specify.

Service additional information URI:

if the service is based on a QC, the Annex L (SD) specifications provide this fields to add a specific URI that supply information about the kind of qualified service offered

Service additional information string:

as above, this string permits to add information about the qualified service

Expired certificate information:

If the current status of the service is suspended or revoked, this date permits to specify the starting time of this status.

Qualifications:

inside the field is indicated, through the “TRUE/FALSE” text, if a QualificationElement [2] is present as extension for this service. See below for a description of a Qualifier section’s contents.

3.8 Service history information section

The service history section can be used to enter information related to audit events related to service.

Service Status Information

*ServiceStatus

*StatusStartingTime

Service Information Extension

Service Information Extension present: TRUE

Apply

Service status: through the drop down can be selected the current state related to the event to be inserted

Status starting time: the starting time of this state

Service information extension: inside the field is indicated, through the “TRUE/FALSE” text, if a QualificationElement [2] is present as extension for this service history instance.

3.9 Qualification Element section

This section accommodate the new format, described in [2], for the service extensions, mandatory for specifying the service characteristics if using QC certificate. As for the preceding information, the value can be inserted through the dialog illustrated below, containing all the fields defined by the schema.

Qualification element

Qualifiers

Available Qualifiers

Add

Currently in this Qualification Element

Remove selected

Criteria List

Assert attribute

☐ Assert all
☐ Assert none
☐ Assert at least one

Key usage

☐ Digital Signature

☐ Key CertSign

☐ Key encipherment

☐ Non repudiation

☐ Decipher only

☐ Encipher only

☐ CRL signature

☐ Data encipherment

☐ Key agreement

Refresh/Add this Key Usage attribute

Remove the KeyUsage attribute

Policy identifier

Identifiers
☐ OIdAsURI
☐ OIdAsURN

Casella di testo di esempio

Add

Description

Casella di testo di esempio

Add

Documentation reference (URI)

Casella di testo di esempio

Add

Currently in this Criteria List

Remove

Remove

Remove

Nested criteria list

Add nested criteria list

Review current

Remove nested CriteriaList

View XML preview

Apply

Qualifiers: the “short” Qualifiers that indicate for which type of service the information are inserted. Pressing “Add” the “full” Qualifier URI will be added to the list on the left and in the XML document

Assert Attribute: this button group define how the KeyUsage attributes is added inside the TSL document, see below

Key Usage: the description of the purposes for which the certificate key is intended to be used. Depending on the Assert attribute value, the KeyUsage attribute will be inserted listing with a node value of “true/false” all the “checked/unchecked” button (Assert All), listing with a node value of true only the checked attribute (Assert at least one) or it will be not inserted (Assert None)

Identifiers: an identifier of the certificate policy. As defined in the XAdES specifications [5], the Identifier can be inserted as URI or as URN, specifying the format with the available check button.

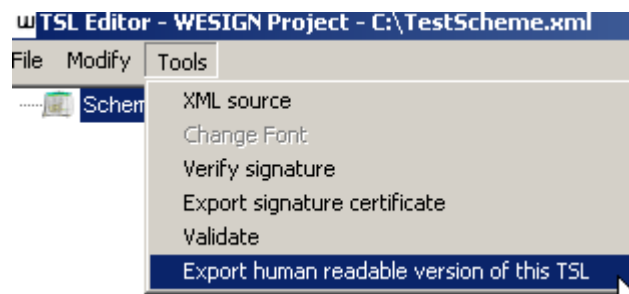
Description: a string that describe the Identifier

Documentation Reference: an URI pointing to documentation reference for the Identifiers inserted.

Pressing the button “Add nested Criteria List” another modal (i.e. floating, not inserted in the main window) pop-up window will appear: inside this window is possible to insert another “CriteriaList” set of attribute to further specify with more attributes the type of service described in the TSL.

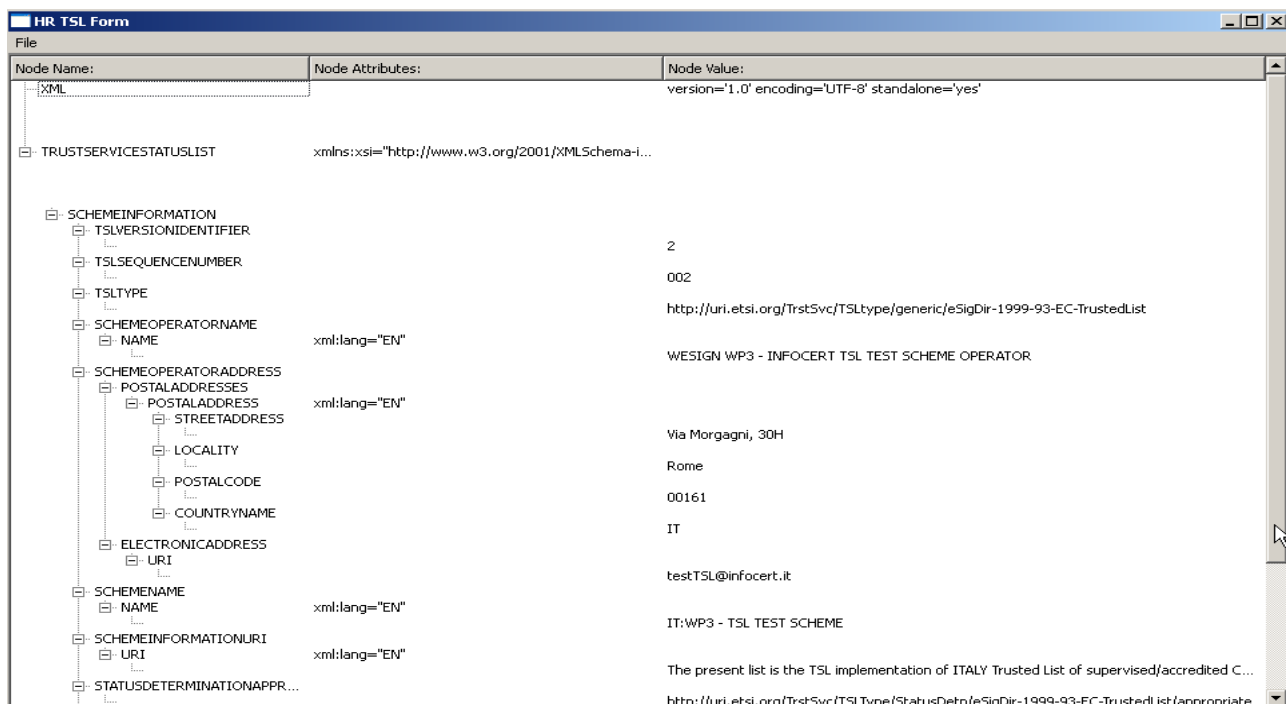
4 TSLExport

To export the TSL in a human-readable form, select from the “Tools” window menu the corresponding entry, as illustrated below.



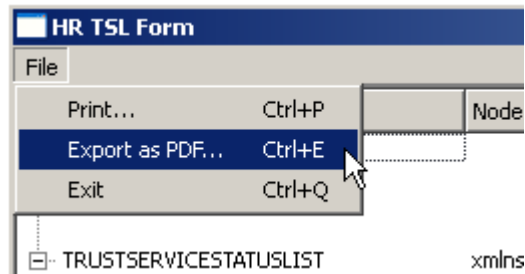
When required, save the TSL file on the local machine. **Careful!** The exporting functions will generate an intermediate html file before the pdf exporting. If you save this “working copy” of the TSL on the “original” file, the xml file will be lost!

After that, an external application will start another window that will show the TSL in a human-readable form.



The window is divided into three columns, holding respectively the node name, the attribute of the node and the node content. The columns are resizable to easier the content reading.

The window menu offer also the possibility to print directly this format of the TSL or to export it in a pdf file.



To fully explode the certificate attribute contents as required by the SD specifications, the TSL as to be exported to a pdf file. Inside the certificate attribute fields will be reported the attribute required for each certificate inserted.

5 Signature and validation

5.4 Signature and validation features

Warning: remember that to use this functions the .NET v2 framework correctly installed is needed

The signature is applied by a little executable file contained in the application folder, XMLEnvSig.exe, called by the TSLEdit application itself.

Once signed the TSL file, built with the editor, the application will not reopen the newly signed file, as the user can't edit it anymore or the signature will become invalid.

All the signed files, if all the procedure completed correctly, will be placed in the "SignedOutput" folder, placed under the TSLEdit.exe folder with their original names plus the suffix "_signed.xml".

5.5 Signing a file step-by-step

The signature function are available under the "File" menu:



Clicking on the menu entry will start the signature process.

5.5.3 PKCS12

- Save the current file.
- Select "PKCS12 – Certificate on file"
- A file dialog will ask for the PKCS12 file.
- A PIN / Password dialog will appear to the user. Enter the correct password to access the private key on the p12 file.
- Wait for the result dialog to appear
- If the result is positive, the signed XML can be found in the "SignedOutput" folder

5.6 Verifying a file

The signature verification can be made selecting the appropriate field under the "Tools" menu.



If the file currently opened has the extension “p7m”, the TSLEdit application will perform a verification directly on it. If the file opened has a simple “xml” extension a dialog will appear asking which file to verify.

After selecting the file that has to be verified, a second file-dialog window will appear: in this window the user can (optionally) select a “trusted” certificate that during the signature verification process will be compared with the one that has signed the TSL. Pressing “Cancel” on this second window allows the user to perform only the integrity check.

The result of the verification, presented to the user through a dialog window, will be composed of two outcomes: the result of the integrity check over the signature (i.e. if the TSL has been altered after the signature or not) and the result of the certificate comparison (match/non-match).

5.7 Validate a file

As shown above in the last image, in the tools menu also a “Validate” function is present. Through this menu function the user can perform a validation of the TSL against the predefined xsd schemes. After the process, if some errors have been found, a dialog box will appear to the user: selecting yes the user can view the validation’s errors and warning list in a dialog box. Otherwise, they can be reviewed opening the temporary text file, named “results.tmp”, created by the validation process inside the application path.

6 Technical notes

The core of the graphical tool is made using the MFC (Microsoft Foundation Classes) framework. However, most of the functionalities offered are performed using the following open source libraries:

- Openssl 0.9.7e
- Libxml2 2.6.11
- Qt4 – Open Source Edition

The tool has been realized during the course of the eTen project WESIGN. The source code has been released as OOSS.

Contact testTSL@infocert.it or info@infocert.it for any further information.

Made by **Infocert s.p.a**



and **Chambersign**



Bibliography

- [1] **ETSI TS 102 231 V3.1.1 Provision of harmonized Trust-service status information**
(2009-06) *Technical Specification - Electronic Signatures and Infrastructures (ESI)*;
- [2] **Technical specifications for a Common Template for the “Trusted List of supervised/ accredited Certification Service Providers”**
- [3] *WESIGN deliverable 3.1 – D3.1 TSL Policy Mapping – available on the private project's website*
- [4] *WESIGN deliverable 3.2 – D3.2 Implementation of TSL Management models – available on the private project's website*
- [5] **ETSI TS 101 903 V1.2.2 XML Advanced Electronic Signature (XAdES)**