

S1000D tools

Documentation

S1000DTOOLS-KHZAE-00000-00
Issue No. 026, 2018-02-26

Publisher:
khzae.net

Produced by: khzae.net

Table of contents

The listed documents are included in issue 026, dated 2018-02-26, of this publication.

Document title	Document identifier	Issue date	No. of pages	Applicable to
Front matter				
Title page	S1000DTOOLS-A-00-00-00-00A-001A-D	2018-02-26	1	All
Table of contents	S1000DTOOLS-A-00-00-00-00A-009A-D	2018-02-16	2	All
Highlights	S1000DTOOLS-A-00-00-00-00A-00UA-D	2018-02-26	1	All
List of abbreviations	S1000DTOOLS-A-00-00-00-00A-005A-D	2017-10-10	1	All
Introduction				
S1000D tools - Description	S1000DTOOLS-A-00-00-00-00A-040A-D	2017-08-14	1	All
Tools for generating data				
s1kd-dmrl - Description	S1000DTOOLS-A-00-0M-00-00A-040A-D	2018-02-23	2	All
s1kd-newcom - Description	S1000DTOOLS-A-00-0G-00-00A-040A-D	2018-02-23	2	All
s1kd-newddh - Description	S1000DTOOLS-A-00-0H-00-00A-040A-D	2018-02-23	2	All
s1kd-newdm - Description	S1000DTOOLS-A-00-07-00-00A-040A-D	2018-02-24	6	All
s1kd-newdml - Description	S1000DTOOLS-A-00-0L-00-00A-040A-D	2018-02-23	2	All
s1kd-newimf - Description	S1000DTOOLS-A-00-0D-00-00A-040A-D	2018-02-23	2	All
s1kd-newpm - Description	S1000DTOOLS-A-00-0C-00-00A-040A-D	2018-02-23	2	All
Tools for authoring and managing data				
s1kd-addicn - Description	S1000DTOOLS-A-00-0R-00-00A-040A-D	2018-02-22	2	All

Document title	Document identifier	Issue date	No. of pages	Applicable to
s1kd-dmls - Description	S1000DTOOLS-A-00-06-00-00A-040A-D	2018-02-22	2	All
s1kd-dmref - Description	S1000DTOOLS-A-00-08-00-00A-040A-D	2018-02-22	2	All
s1kd-metadata - Description	S1000DTOOLS-A-00-09-00-00A-040A-D	2018-02-23	2	All
s1kd-transform - Description	S1000DTOOLS-A-00-0F-00-00A-040A-D	2018-02-23	2	All
s1kd-upissue - Description	S1000DTOOLS-A-00-05-00-00A-040A-D	2018-02-23	3	All
Tools for validating data				
s1kd-brexcheck - Description	S1000DTOOLS-A-00-04-00-00A-040A-D	2018-02-23	4	All
s1kd-checkrefs - Description	S1000DTOOLS-A-00-0J-00-00A-040A-D	2018-02-22	2	All
s1kd-refls - Description	S1000DTOOLS-A-00-0P-00-00A-040A-D	2018-02-23	2	All
s1kd-validate - Description	S1000DTOOLS-A-00-02-00-00A-040A-D	2018-02-23	2	All
Tools for delivering data				
s1kd-acronyms - Description	S1000DTOOLS-A-00-0K-00-00A-040A-D	2018-02-22	2	All
s1kd-flatpm - Description	S1000DTOOLS-A-00-0N-00-00A-040A-D	2018-02-23	2	All
s1kd-instance - Description	S1000DTOOLS-A-00-03-00-00A-040A-D	2018-02-26	8	All
s1kd-neutralize - Description	S1000DTOOLS-A-00-0E-00-00A-040A-D	2018-02-23	2	All
s1kd-syncrefs - Description	S1000DTOOLS-A-00-01-00-00A-040A-D	2018-02-23	1	All

Highlights

The listed changes are introduced in issue 026, dated 2018-02-26, of this publication.

Data module code	Reason for update
S1000DTOOLS-A-00-01-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-02-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-03-00-00A-040A-D	Provide a set of built-in XSLT scripts for resolving CIR items. Clarify custom XSLT example.
S1000DTOOLS-A-00-04-00-00A-040A-D	Reformat XML examples.
S1000DTOOLS-A-00-05-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-06-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-07-00-00A-040A-D	Correct list of attributes in 'dmtypes' XML file description.
S1000DTOOLS-A-00-08-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-09-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-0C-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-0D-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-0E-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-0F-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-0G-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-0H-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-0J-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-0K-00-00A-040A-D	Improve organization and formatting. Add more detail to description.
S1000DTOOLS-A-00-0L-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-0M-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-0N-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-0P-00-00A-040A-D	Improve organization and formatting.
S1000DTOOLS-A-00-0R-00-00A-040A-D	Improve organization and formatting.

List of abbreviations

BREX	Business Rules EXchange
CIR	Common Information Repository
PCT	Product Cross-reference Table
SNS	Standard Numbering System

S1000D tools

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1

List of tables

1 References.....	1
------------------------	---

References

Table 1 References

Data module/Technical publication	Title
https://github.com/kibook/S1000D-XSL-Stylesheets	S1000D XSL stylesheets
https://github.com/kibook/s1kd-tools	s1kd-tools

Description

1 General

s1kd-tools are a set of small tools for manipulating S1000D data. They are maintained at <https://github.com/kibook/s1kd-tools>.

This publication is meant to serve as an example of an S1000D data set produced using these tools. The stylesheets used to produce this PDF can be found at <https://github.com/kibook/S1000D-XSL-Stylesheets>

s1kd-dmrl

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1

List of tables

1 References.....	1
-------------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 General

The **s1kd-dmrl** tool reads S1000D data management lists and creates CSBD objects for the entries specified using the s1kd-new* tools.

2 Usage

```
s1kd-dmrl [-Nh?] <DML>...
```

3 Options

-F	Fail on the first error generated by any of the s1kd-new* commands. Normally, errors with individual DMRL entries will be reported but the other entries will still be processed.
-f	Overwrite existing CSDB objects.
-h -?	Show help/usage message.

-N	Omit issue/in-work numbers from the filenames of created CSDB objects.
-s	Do not create CSDB objects, only output the s1kd-new* commands to create them.
<DML>...	One or more S1000D data management lists.

s1kd-newcom

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1
3.1 'defaults' file.....	2

List of tables

1 References.....	1
------------------------	---

References

Table 1 References

Data module/Technical publication	Title
S1000DTOOLS-A-00-07-00-00A-040A-D	s1kd-newdm - Description

Description

1 **General**

The **s1kd-newcom** tool creates a new S1000D comment with the code and metadata specified.

2 **Usage**

s1kd-newcom [options]

3 **Options**

-# <code>

The code of the comment, in the form of
MODELIDENTCODE-SENDERIDENT-YEAR-SEQ-TYPE.

-\$ <issue>

Specify which issue of S1000D to use. Currently supported
issues are:

- 4.2 (default)
- 4.1

	– 4.0
	– 3.0
	– 2.3
-@ <filename>	Save the new comment as <filename> instead of an automatically named file in the current directory.
-% <dir>	Use the XML template in the specified directory instead of the built-in template. The template must be named <code>comment.xml</code> inside <dir> and must conform to the default S1000D issue (4.2).
-b <BREX>	BREX data module code.
-C <country>	The country ISO code of the new comment.
-c <sec>	The security classification of the new comment.
-d <defaults>	Specify the 'defaults' file name.
-f	Overwrite existing file.
-l <date>	The issue date of the new comment in the form of YYYY-MM-DD.
-L <lang>	The language ISO code of the new comment.
-o <orig>	The enterprise name of the originator of the comment.
-p	Prompt the user for values left unspecified.
-r <type>	The response type of the new comment.
-t <title>	The title of the new comment.
-v	Print the file name of the newly created comment.

3.1 'defaults' file

Refer to [S1000DTOOLS-A-00-07-00-00A-040A-D](#) for information on the 'defaults' file which is used by all the s1kd-new* commands.

s1kd-newddn

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1
3.1 'defaults' file.....	2

List of tables

1	References.....	1
---	-----------------	---

References

Table 1 References

Data module/Technical publication	Title
S1000DTOOLS-A-00-07-00-00A-040A-D	s1kd-newdm - Description

Description

1 General

The **s1kd-newddn** tool creates a new S1000D data dispatch note with the code, metadata, and list of files specified.

2 Usage

```
s1kd-newddn [options] <files>...
```

3 Options

-# <code>

The code of the new data dispatch note, in the form of MODELIDENTCODE-SENDER-RECEIVER-YEAR-SEQUENCE.

-\$ <issue>

Specify which issue of S1000D to use. Currently supported issues are:

	– 4.2 (default)
	– 4.1
	– 4.0
	– 3.0
	– 2.3
-@ <filename>	Save the new DDN as <filename> instead of an automatically named file in the current directory.
-% <dir>	Use the XML template in the specified directory instead of the built-in template. The template must be named <code>ddn.xml</code> inside <dir> and must conform to the default S1000D issue (4.2).
-a <auth>	Specify the authorization.
-b <BREX>	BREX data module code.
-d <defaults>	Specify the 'defaults' file name.
-f	Overwrite existing file.
-h -?	Show help/usage message.
-l <date>	The issue date of the new DDN in the form of YYYY-MM-DD.
-N <country>	The receiver's country.
-n <country>	The sender's country.
-o <sender>	The enterprise name of the sender.
-p <showprompts>	Prompt the user for values left unspecified.
-r <receiver>	The enterprise name of the receiver.
-T <city>	The receiver's city.
-t <city>	The sender's city.
-v	Print the file name of the newly created DDN.

3.1 'defaults' file

Refer to [S1000DTOOLS-A-00-07-00-00A-040A-D](#) for information on the 'defaults' file which is used by all the `s1kd-new*` commands.

s1kd-newdm

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1
3.1 Prompt (-p) option.....	3
3.2 'defaults' file.....	3
3.3 'dmtypes' file.....	5
3.4 Custom XML templates (-%).....	6
4 Example.....	6

List of tables

1	References.....	1
---	-----------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 General

The **s1kd-newdm** tool creates a new S1000D data module with the data module code and other metadata specified.

2 Usage

```
s1kd-newdm [options]
```

3 Options

-# <DMC> The data module code of the new data module.

-\$ <issue> Specify which issue of S1000D to use. Currently supported issues are:

	– 4.2 (default)
	– 4.1
	– 4.0
	– 3.0
	– 2.3
-@ <filename>	Save the new data module as <filename> instead of an automatically named file in the current directory.
-% <dir>	Use XML templates in the specified directory instead of the built-in templates.
-,	Dumps the built-in default 'dmotypes' XML. This can be used to quickly set up a starting point for a project's custom info codes, from which info names can be modified and unused codes can be removed to fit the project.
-.	Dumps the simple text form of the built-in default 'dmotypes'.
-b <BREX>	BREX data module code.
-C <country>	The country ISO code of the new data module.
-c <sec>	The security classification of the new data module.
-D <dmotypes>	Specify the 'dmotypes' file name.
-d <defaults>	Specify the 'defaults' file name.
-f	Overwrite existing file.
-I <date>	Issue date of the new data module in the form of YYYY-MM-DD.
-i <info>	The info name of the new data module.
-L <language>	The language ISO code of the new data module.
-m <remarks>	Set remarks for the new data module.
-N	Omit issue/inwork numbers from filename.
-n <issue>	The issue number of the new data module.
-O <CAGE>	The CAGE code of the originator.
-o <orig>	The originator enterprise name of the new data module.
-p	Prompts the user for any values left unspecified.
-R <CAGE>	The CAGE code of the responsible partner company.
-r <RPC>	The responsible partner company enterprise name of the new data module.
-S <BREX>	Determine the tech name from the SNS rules of a specified BREX data module. This can also be specified in the 'defaults' file with the key 'sns'.
-s <schema>	The schema URL.

-T <schema>	<p>The type (schema) of the new data module. Supported schemas:</p> <ul style="list-style-type: none">– appliccrossreftable - Applicability cross-reference table– brdoc - Business rule document– brex - Business rule exchange– checklist - Maintenance checklist– comrep - Common information repository– condcrossreftable - Conditions cross-reference table– container - Container– crew - Crew/Operator information– descript - Descriptive– fault - Fault information– frontmatter - Front matter– ipd - Illustrated parts data– learning - Technical training information– prdcrossreftable - Product cross-reference table– proced - Procedural– process - Process– sb - Service bulletin– schedul - Maintenance planning information– scocontent - SCO content information– techrep - Technical repository (replaced by comrep in issue 4.1)– wrngdata - Wiring data– wrngflds - Wiring fields
-t <tech>	The tech name of the new data module.
-v	Print the file name of the newly created data module.
-w <inwork>	The inwork number of the new data module.

3.1 Prompt (-p) option

If this option is specified, the program will prompt the user to enter values for metadata which was not specified when calling the program. If a piece of metadata has a default value (from the 'defaults' and 'dmtypes' files), it will be displayed in square brackets [] in the prompt, and pressing Enter without typing any value will select this default value.

3.2 'defaults' file

This file sets default values for each piece of metadata. By default, the program will search the current directory for a file named 'defaults', but any file can be specified by using the -d option.

All of the s1kd-new* commands use the same 'defaults' file format, so this file can contain default values for multiple types of metadata.

Each line consists of the identifier of a piece of metadata and its default value, separated by whitespace. Lines which do not match a piece of metadata are ignored, and may be used as comments. Example:

```
# General
modelIdentCode          S1000DTOOLS
securityClassification   01
responsiblePartnerCompany khzae.net
originator              khzae.net
languageIsoCode         en
countryIsoCode          CA
issueNumber             000
inWork                  01

# Data modules
systemDiffCode          A
systemCode              00
subSystemCode           0
subSubSystemCode        0
assyCode                00
disassyCode             00
disassyCodeVariant      A
infoCode                040
infoCodeVariant         A
itemLocationCode        D

# Comments/DDN
senderIdent             KHZAE
yearOfDataIssue         2017
seqNumber               00001
city                   Toronto
country                 Canada

# Comments
commentType             q
commentPriorityCode      cp01

# DDN
authorization           khzae.net

# Publication modules
pmIssuer                KHZAE
pmNumber                00001
pmVolume                00
```


Alternatively, the 'defaults' file can be written using an XML format, containing a root element `defaults` with child elements `default` which each have an attribute `ident` and an attribute `value`.

```
<?xml version="1.0"?>
<defaults>
<!-- General -->
<default ident="modelIdentCode" value="S1000DTOOLS"/>
<default ident="securityClassification" value="01"/>
[...]
</defaults>
```

3.3 'dmtypes' file

This file sets the default type (schema) for data modules based on their info code. By default, the program will search the current directory for a file named 'dmtypes', but any file can be specified by using the -D option.

Each line consists of an info code, a schema identifier, and optionally a default info name. Example:

```
00E    comrep
00W    appliccrossreftable
009    frontmatter
022    brex
024    brdoc
040    descript      Description
520    proced        Remove procedure
```

Like the 'defaults' file, the 'dmtypes' file may also be written in an XML format, where each child has an attribute `infoCode`, an attribute `schema`, and optionally an attribute `infoName`.

```
<?xml version="1.0">
<dmtypes>
<type infoCode="022" schema="brex"/>
<type infoCode="040" schema="descript" infoName="Description"/>
<type infoCode="520" schema="proced" infoName="Remove procedure"/>
</dmtypes>
```

Info code variants can also be given specific default schema and info names. To do this, include the variant with the info code:

```
258A  proced  Other procedure to clean
258B  proced  Other procedure to clean, Clean with air
258C  proced  Other procedure to clean, Clean with water
```

The two forms of info codes (with and without variant) can be mixed. Defaults are chosen in the order they are listed in the 'dmtypes' file. An info code with no variant matches all possible variants.

3.4 Custom XML templates (-%)

A minimal set of S1000D templates are built-in to this tool, but customized templates may be used with the `-%` option. This option takes a path to a directory where the custom templates are located. Each template should be named `<schema>.xml`, where `<schema>` is the name of the schema, matching one of the schema names in the 'dmtypes' file or the schema specified with the `-T` option.

The templates must be written to conform to the default S1000D issue of this tool (currently 4.2). They will be automatically transformed when another issue is specified with the `-$` option.

The 'templates' default can also be specified in the 'defaults' file to use these custom templates by default.

4 Example

```
$ slkd-newdm -# S1000DTOOLS-A-00-07-00-00A-040A-D
```

s1kd-newdml

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1
3.1 'defaults' file.....	2

List of tables

1	References.....	1
---	-----------------	---

References

Table 1 References

Data module/Technical publication	Title
S1000DTOOLS-A-00-07-00-00A-040A-D	s1kd-newdml - Description

Description

1 General

The **s1kd-newdml** tool creates a new S1000D data management list with the code and other metadata specified.

2 Usage

```
s1kd-newdml [options] [<datamodules>]
```

3 Options

- | | |
|-------------|---|
| -# <code> | The data management list code of the new DML. |
| -\$ <issue> | Specify which issue of S1000D to use. Currently supported issues are: |
| | – 4.2 (default) |
| | – 4.1 |

	- 4.0
	- 3.0
	- 2.3
-@ <filename>	Save new DML to <filename> instead of an automatically named file in the current directory.
-% <dir>	Use the XML template in the specified directory instead of the built-in template. The template must be named <code>dml.xml</code> inside <dir> and must conform to the default S1000D issue (4.2).
-b <BREX>	BREX data module code.
-c <sec>	The security classification of the new data module.
-d <defaults>	Specify the 'defaults' file name.
-f	Overwrite existing file.
-h -?	Show usage message.
-l <date>	The issue date of the new DML in the form of YYYY-MM-DD.
-N	Omit the issue/inwork numbers from filename.
-n <issue>	The issue number of the new data module.
-p	Prompts the user for any values left unspecified.
-R <NCAGE>	Specifies a default responsible partner company enterprise code for entries which do not carry this in their ID STATUS section (ICN, COM, DML).
-r <name>	Specifies a default responsible partner company enterprise name for entries which do not carry this in their IDSTATUS section (ICN, COM, DML).
-v	Print the file name of the newly created DML.
-w <inwork>	The inwork number of the new data module.
<datamodules>	Any number of data module file names to automatically add to the list.

3.1 'defaults' file

Refer to [S1000DTOOLS-A-00-07-00-00A-040A-D](#) for information on the 'defaults' file which is used by all the s1kd-new* commands.

s1kd-newimf

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1

List of tables

1 References.....	1
------------------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 **General**

The **s1kd-newimf** tool creates a new S1000D ICN metadata file for specified ICN files.

2 **Usage**

```
s1kd-newimf [options] <ICNs>...
```

3 **Options**

-% <dir>	Use the XML template in <dir> instead of the built-in template. The template must be named <code>icnmetadata.xml</code> inside <dir> and must conform to the default S1000D issue (4.2).
-b <BREX>	BREX data module code.
-c <sec>	The security classification of the new ICN metadata file.
-d <defaults>	Specify the 'defaults' file name.

-f	Overwrite existing file.
-l <date>	The issue date of the new ICN metadata file in the form of YYYY-MM-DD.
-n <issue>	The issue number of the new ICN metadata file.
-O <CAGE>	The CAGE code of the originator.
-o <orig>	The originator enterprise name of the new ICN metadata file.
-p	Prompts the user for any values left unspecified.
-R <CAGE>	The CAGE code of the responsible partner company.
-r <RPC>	The responsible partner company enterprise name of the new ICN metadata file.
-t <title>	The ICN title (if creating multiple ICNs, they will all use this title).
-v	Print the file name of the newly created IMF.
-w <inwork>	The inwork issue of the new ICN metadata file.

s1kd-newpm

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1
3.1 'defaults' file.....	2

List of tables

1	References.....	1
---	-----------------	---

References

Table 1 References

Data module/Technical publication	Title
S1000DTOOLS-A-00-07-00-00A-040A-D	s1kd-newdm - Description

Description

1 General

The **s1kd-newpm** tool creates a new S1000D publication module with the publication module code and other metadata specified.

2 Usage

s1kd-newpm [options]

3 Options

-# <PMC>

The publication module code of the new publication module.

-\$ <issue>

Specify which issue of S1000D to use. Currently supported issues are:

- 4.2 (default)
- 4.1

	- 4.0
	- 3.0
	- 2.3
-@ <filename>	Save new publication module as <filename> instead of an automatically named file in the current directory.
-% <dir>	Use the XML template in <dir> instead of the built-in template. The template must be named <code>pm.xml</code> in <dir> and must conform to the default S1000D issue (4.2).
-b <BREX>	BREX data module code.
-C <country>	The country ISO code of the new publication module.
-c <sec>	The security classification of the new publication module.
-d	Specify the 'defaults' file name.
-f	Overwrite existing file.
-l <date>	The issue date of the new publication module in the form of YYYY-MM-DD.
-L <language>	The language ISO code of the new publication module.
-n <issue>	The issue number of the new publication module.
-p	Prompt the user for any values left unspecified.
-R <CAGE>	The CAGE code of the responsible partner company.
-r <RPC>	The responsible partner company enterprise name of the new publication module.
-t <title>	The title of the new publication module.
-v	Print the file name of the newly created publication module.
-w <inwork>	The inwork number of the new publication module.

3.1 'defaults' file

Refer to [S1000DTOOLS-A-00-07-00-00A-040A-D](#) for information on the 'defaults' file which is used by all the `s1kd-new*` commands.

s1kd-addicn

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1

List of tables

1 References.....	1
------------------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 **General**

The **s1kd-addicn** tool adds the required DTD entity and notation declarations to an S1000D module in order to reference an ICN file.

2 **Usage**

```
s1kd-addicn [-s <src>] [-o <out>] [-fh?] <ICN>...
```

3 **Options**

-F	Use the whole path given for the ICN file as the SYSTEM ID.
-f	Overwrite source file instead of writing to stdout.
-h -?	Show help/usage message.
-o <out>	The filename to output to. Default is to write to stdout.
-s <src>	The source module to add the ICN(s) to. Default is to read from stdin.

<ICN>..

Any number of ICN files to add.

s1kd-dmls

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1

List of tables

1 References.....	1
------------------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 **General**

The **s1kd-dmls** tool lists data modules in a directory, with various options for columns for data module metadata which can be useful for sorting them with other tools.

2 **Usage**

```
s1kd-dmls [-acfHhilorTtpDP]
```

3 **Options**

-a	Include the applicability column.
-c	Show data module code column.
-D, -P, -C, -M	List data modules, publication modules, comments and ICN metadata files respectively. If none are specified, -DPCM is assumed.
-f	Do not show filename column.

-H	Show headers on columns.
-h -?	Show the usage message.
-l	Show only official issues of data modules (inwork = 00).
-i	Include the issue date column.
-L	Show language info (languageIsoCode-countryIsoCode).
-l	Show only the latest issue/inwork version of data modules.
-n	Show issue info (issueNumber-inWork).
-o	Include the originator column.
-p	Do not replace control characters (\n, \t) when printing.
-R	Recursively descend in to directories.
-r	Include the responsible partner company column.
-T	Show title in single column (techName - infoName).
-t	Show tech and info name columns.
-w	Show only writable data module files.

s1kd-dmref

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1
4 Example.....	2

List of tables

1 References.....	1
------------------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 **General**

The **s1kd-dmref** tool generates the XML for a <dmRef> element using the specified code or data module filename. When using a filename, it can parse the data module to include the issue, language, and/or title information in the reference.

2 **Usage**

```
s1kd-dmref [-tlih?] [<code>|<filename>]
```

3 **Options**

-h -?	Show the usage message.
-i	Include the issue information in the reference (target must be a file)
-l	Include the language information in the reference (target must be a file)

-t Include the dmTitle in the reference (target must be a file).
<code>|<filename> Either a data module code, including the prefix DMC or DME (for extended identification), or the filename of a data module.

4 Example

```
$ slkd-dmref DMC-S1000DTOOLS-A-00-08-00-00A-040A-D
<dmRef>
<dmRefIdent>
<dmCode modelIdentCode="S1000DTOOLS" systemDiffCode="A"
systemCode="00" subSystemCode="0" subSubSystemCode="8" assyCode="00"
disassyCode="00" disassyCodeVariant="A" infoCode="040"
infoCodeVariant="A" itemLocationCode="D"/>
</dmRefIdent>
</dmRef>
```

s1kd-metadata

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1
4 Example.....	2

List of tables

1	References.....	1
---	-----------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1**General**

The **s1kd-metadata** tool provides a simple way to fetch and change metadata on S1000D data modules.

2**Usage**

```
s1kd-metadata [-c <file>] [-tf] [-n <name> [-v <value>]] [<module>]
```

3**Options**

-c <file>

Use <file> to edit metadata files. <file> consists of lines starting with a metadata name, followed by whitespace, followed by the new value for the metadata (the program uses this same format when outputting all metadata if no <name> is specified).

-f	When editing metadata, overwrite the module. The default is to output the modified module to stdout.
-n <name>	The name of the piece of metadata to fetch. If no name is specified, all available metadata names are printed with their values. This output can be sent to a text file, edited, and then specified with the -c option as a means of editing metadata in any text editor.
-t	Do not format columns in output.
-v <value>	The new value for the piece of metadata specified by -n.
<module>	The module to show/edit metadata on. The default is to read from stdin.

4 Example

```
$ ls
DMC-S1000DTOOLS-A-00-09-00-00A-040A-D_EN-CA.XML

$ slkd-metadata DMC-S1000DTOOLS-A-00-09-00-00A-040A-D_EN-CA.XML
issueDate                2017-08-14
techName                  slkd-metadata(1) | slkd-tools
responsiblePartnerCompany khzae.net
originator                khzae.net
securityClassification    01
schema                   http://www.s1000d.org/S1000D_4-2/xml_
schema_flat/descript.xsd
type                      dmodule
applic                    All
brex                      S1000D-F-04-10-0301-00A-022A-D
issueType                 new
language                  en-CA
issueInfo                  001-00
dmCode                    S1000DTOOLS-A-00-09-00-00A-040A-D

$ slkd-metadata DMC-S1000DTOOLS-A-00-09-00-00A-040A-D_EN-CA.XML \
-n techName -v 'New title'
$ slkd-metadata DMC-S1000DTOOLS-A-00-09-00-00A-040A-D_EN-CA.XML \
-n techName
New title
```


s1kd-transform

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1
3.1 Identity template.....	2

List of tables

1	References.....	1
---	-----------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 General

Applies an XSLT stylesheet to S1000D data modules. The DTD of any specified data modules is preserved in the resulting output.

2 Usage

```
s1kd-transform [-s <stylesheet> ...] [-o <file>] [-ifh?] [<data module> ...]
```

3 Options

-f	Overwrite the specified data module(s) instead of writing to stdout.
-h -?	Show usage message.
-i	Includes an "identity" template in to each specified stylesheet.

-o <file>	Output to <file> instead of stdout. This option only makes sense when the input is a single data module.
-s <stylesheet>	An XSLT stylesheet file to apply to each data module. Multiple stylesheets can be specified by supplying this argument multiple times. The stylesheets will be applied in the order they are listed.
<data module> ...	Any number of data modules to apply all specified stylesheets to.

3.1 Identity template

The -i option includes an "identity" template in to each stylesheet specified with the -s option. The template is equivalent to this XSL:

```
<xsl:template match="@*|node()">
<xsl:copy>
<xsl:apply-templates select="@*|node()" />
</xsl:copy>
</xsl:template>
```

This means that any attributes or nodes which are not matched by a more specific template in the user-specified stylesheet are copied.

s1kd-upissue

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1
4 Examples.....	2
4.1 Data module with issue/inwork in filename.....	2
4.2 Data module without issue/inwork in filename.....	2
4.3 Non-XML file with issue/inwork in filename.....	3

List of tables

1	References.....	1
---	-----------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 General

The **s1kd-upissue** tool increases the in-work or issue number of an S1000D data module, publication module, etc.

Any files using an S1000D-esque naming convention, placing the issue and in-work numbers after the first underscore (_) character, can also be "upissued". Files which do not contain the appropriate S1000D metadata are simply copied.

2 Usage

```
s1kd-upissue [-viNrQI] [-1 <type>] [-2 <type>] [-s <status>] <files>
```

3 Options

-1 <type> Set first verification type (tabtop, onobject, ttandoo).

-2 <type>	Set second verification type (tabtop, onobject, ttandoo).
-l	Do not change issue date. Normally, when upissuing to the next inwork or official issue, the issue date is changed to the current date. This option will keep the date of the previous inwork or official issue.
-i	Increase the issue number of the data module. By default, the in-work issue is increased.
-N	Omit issue/inwork numbers from filename.
-q	Keep quality assurance information from old issue. Normally, when upissuing an official data module to the first in-work issue, the quality assurance is set back to "unverified". Specify this option to indicate the upissue will not affect the contents of the data module, and so does not require it to be re-verified.
-R	Delete only change markup on elements associated with an RFU (by use of the attribute <code>reasonForUpdateRefIds</code>). Change markup on other elements is ignored.
-r	Keep old RFUs. Normally, when upissuing an official data module to the first in-work issue, any reasons for update are deleted automatically, along with any change markup attributes on elements. This option prevents their deletion.
-s <status>	Set the status of the new issue. Default is 'changed'.
-v	Print the file name of the upissued data module.

4 Examples

4.1 Data module with issue/inwork in filename

```
$ ls
DMC-S1000DTOOLS-A-00-00-00-00A-040A-D_000-01_EN-CA.XML

$ slkd-upissue DMC-S1000DTOOLS-A-00-00-00-00A-040A-D_000-01_EN-CA.XML
$ ls
DMC-S1000DTOOLS-A-00-00-00-00A-040A-D_000-01_EN-CA.XML
DMC-S1000DTOOLS-A-00-00-00-00A-040A-D_000-02_EN-CA.XML

$ slkd-upissue \
-i DMC-S1000DTOOLS-A-00-00-00-00A-040A-D_000-02_EN-CA.XML
$ ls
DMC-S1000DTOOLS-A-00-00-00-00A-040A-D_000-01_EN-CA.XML
DMC-S1000DTOOLS-A-00-00-00-00A-040A-D_000-02_EN-CA.XML
DMC-S1000DTOOLS-A-00-00-00-00A-040A-D_001-00_EN-CA.XML
```

4.2 Data module without issue/inwork in filename

```
$ ls
```

DMC-S1000DTOOLS-A-00-00-00-00A-040A-D_EN-US.XML

```
$ slkd-metadata DMC-S1000DTOOLS-A-00-00-00-00A-040A-D_EN-CA.XML \  
-n issueInfo  
000-01  
$ slkd-upissue -N DMC-S1000DTOOLS-A-00-00-00-00A-040A-D_EN-CA.XML  
$ slkd-metadata DMC-S1000DTOOLS-A-00-00-00-00A-040A-D_EN-CA.XML \  
-n issueInfo  
000-02
```

4.3 Non-XML file with issue/inwork in filename

```
$ ls  
TXT-S1000DTOOLS-KHZAE-FOOBAR_000-01_EN-CA.TXT  
  
$ slkd-upissue TXT-S1000DTOOLS-KHZAE-00001_000-01_EN-CA.TXT  
$ ls  
TXT-S1000DTOOLS-KHZAE-FOOBAR_000-01_EN-CA.TXT  
TXT-S1000DTOOLS-KHZAE-FOOBAR_000-02_EN-CA.TXT
```

s1kd-brexcheck

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1
3.1 Business rule severity levels (-w).....	2
3.2 Normal, strict and unstrict SNS check (-S, -St, -Su).....	3
4 Return value.....	4
5 Example.....	4

List of tables

1	References.....	1
---	-----------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 General

The **s1kd-brexcheck** tool validates an S1000D data module using the context rules, SNS rules, and/or notation rules of one or multiple BREX (Business Rules EXchange) data modules. All errors are displayed with the <objectUse> message, the line number, and a representation of the invalid XML tree.

2 Usage

```
s1kd-brexcheck [-b <brex>] [-I <path>] [-w <severities>]
               [-vVqDsxlStupfch?] <datamodules>
```

3 Options

-b <brex> Check the data modules against this BREX. Multiple BREX data modules can be specified by adding this option multiple

	times. When no BREX data modules are specified, the BREX data module referenced in <brexDmRef> in the data module is attempted to be used instead.
-c	When a context rule defines values for an object (objectValue), check if the value of each object is within the allowed set of values.
-f	Output only the filenames of modules with BREX/SNS errors.
-h -?	Show the help/usage message.
-l <path>	Add a search path for BREX data modules. By default, only the current directory is searched.
-l	Use the layered BREX concept. BREX data modules referenced by other BREX data modules (either specified with -b or referenced by the specified data modules) will also be checked against.
-n	Check notation rules. Any notation names listed in any of the BREX data modules with attribute allowedNotationFlag set to "1" or omitted are considered valid notations. If a notation in a data module is not present or has allowedNotationFlag set to "0", an error will be returned. For notations not included but not explicitly excluded, the objectUse of the first inclusion rule will be returned with the error. For explicitly excluded notations, the objectUse of the explicit exclusion rule is returned.
-p	Display a progress bar.
-S[tu]	Check SNS (Standard Numbering System) rules. The SNS of each specified data module is checked against the combination of all SNS rules of all specified BREX data modules.
-s	Use shortened, single-line messages to report BREX errors instead of multiline indented messages.
-v -V -q -D	Verbosity of the output.
-w <severities>	Specify a list of severity levels for business rules.
-x	Output an XML report instead of a plain-text one.

3.1 Business rule severity levels (-w)

The attribute brSeverityLevel on a BREX rule allows for distinguishing different kinds of errors. The -w option takes an XML file containing a list of severity levels, their user-defined type, and optionally if they should not be counted as true errors (causing the tool to return a "failure" status) but merely warnings.

An example of the format of this file is given below:

```
<?xml version="1.0"?>
<brSeverityLevels>
```

```
<brSeverityLevel value="brsl01" fail="yes">Error</brSeverityLevel>
<brSeverityLevel value="brsl02" fail="no">Warning</brSeverityLevel>
</brSeverityLevels>
```

When the attribute `fail` has a value of "yes" (or is not included), BREX errors pertaining to rules with the given severity level value will be counted as errors. When it is no, the errors are still displayed but are not counted as errors in the exit status code of the tool.

3.2 Normal, strict and unstrict SNS check (-S, -St, -Su)

There are three modes for SNS checking: normal, strict, and unstrict. The main difference between them is how they handle the optional levels of an SNS description in the BREX.

-St enables **strict** SNS checking. By default, the normal SNS check (-S) will assume optional elements `snsSubSystem`, `snsSubSubSystem`, and `snsAssy` exist with an `snsCode` of "0" ("00" or "0000" for `snsAssy`) when their parent element does not contain any of each. This provides a shorthand, such that

```
<snsSystem>
<snsCode>00</snsCode>
<snsTitle>General</snsTitle>
</snsSystem>
```

is equivalent to

```
<snsSystem>
<snsCode>00</snsCode>
<snsTitle>General</snsTitle>
<snsSubSystem>
<snsCode>0</snsCode>
<snsTitle>General</snsTitle>
<snsSubSubSystem>
<snsCode>0</snsCode>
<snsTitle>General</snsTitle>
<snsAssy>
<snsCode>00</snsCode>
<snsTitle>General</snsTitle>
</snsAssy>
</snsSubSubSystem>
</snsSubSystem>
</snsSystem>
```

Using strict checking will disable this shorthand, and missing optional elements will result in an error.

-Su enables **unstrict** SNS checking. The normal SNS check (-S) shorthand mentioned above only allows SNS codes of "0" to be omitted from the SNS rules. Using unstrict checking, **any** code used will not produce an error when the relevant optional elements are omitted. This means that given the following...

```
<snsSystem>
```



```
<snsCode>00</snsCode>
<snsTitle>General</snsTitle>
</snsSystem>
```

...SNS codes of 00-00-0000 through 00-ZZ-ZZZZ are considered valid.

4 Return value

The number of BREX errors encountered is returned in the exit status code.

5 Example

```
$ DMOD=DMC-S1000DTOOLS-A-00-00-00-00A-040A-D_000-01_EN-CA.XML
$ BREX=DMC-S1000D-F-04-10-0301-00A-022A-D_001-00_EN-US.XML
$ cat $DMOD
[...]
<listItem id="stp-0001">
<para>List items shouldn't be used as steps...</para>
</listItem>
[...]
<para>Refer to <internalRef internalRefId="stp-0001"
internalRefTargetType="irtt08"/>.</para>
[...]

$ slkd-brexcheck -b $BREX $DMOD
BREX ERROR: DMC-S1000DTOOLS-A-00-00-00-00A-040A-D_000-01_EN-CA.XML
  Only when the reference target is a step can the value of attribute
  internalRefTargetType be irtt08 (Chap 3.9.5.2.1.2, Para 2.1).
  line 53:
    ELEMENT internalRef
      ATTRIBUTE internalRefId
        TEXT
          content=stp-0001
      ATTRIBUTE internalRefTargetType
        TEXT
          content=irtt08
```

s1kd-checkrefs

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1
3.1 External publication list (-e).....	2

List of tables

1	References.....	1
---	-----------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 General

The **s1kd-checkrefs** tool takes a list of S1000D data modules and pub modules, and lists any invalid references to data/pub modules within them (references to modules not included in the list). It can also update the address items (title, issueDate if applicable) of all valid references using the corresponding address items of the given modules.

2 Usage

```
s1kd-checkrefs [-s <source>] [-t <target>]
               [-d <dir>] [-cuFelvh?] <modules>...
```

3 Options

-c Only check/update references within the content section of modules.

-d <dir>	Check references between data modules in the specified directory. Additional data modules can still be specified with -s.
-e	Check/update external publication references against a pre-defined list of publications.
-F	Fail on first invalid reference and return an error code.
-h -?	Show help/usage message
-l	List all invalid references found.
-s <source>	Use only the specified module as the source of address items. Only references to this module will be checked and/or updated in all other modules.
-t <target>	Only check and/or update references within this module. All other modules will only be used as sources.
-u	Update the address items of all valid references found within the specified modules.
-v	Verbose output.

3.1 External publication list (-e)

Since external publications can be of any format, in order to check references to them, their metadata must be specified in an XML format for the s1kd-checkrefs tool to read.

The root element of the XML file is the `externalPubs` element. Each external publication is represented by an element `externalPubAddress`. The identifying elements of the publication are stored in the `externalPubIdent` element (corresponding with the `externalPubRefIdent` element). The address items are stored in the `externalPubAddressItems` element (corresponding with the `externalPubRefAddressItems` element).

Example:

```
<?xml version="1.0"?>
<externalPubs>
<externalPubAddress>
<externalPubIdent>
<externalPubCode>s1kd-checkrefs</externalPubCode>
<externalPubTitle>s1kd-checkrefs manual</externalPubTitle>
</externalPubIdent>
<externalPubAddressItems>
<externalPubIssueDate year="2017" month="08" day="14"/>
</externalPubAddressItems>
</externalPubAddress>
</externalPubs>
```

s1kd-refls

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1

List of tables

1 References.....	1
------------------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 General

The **s1kd-refls** tool lists external references to other CSDB objects (dmRef, pmRef), optionally matching them to a filename in the current directory. This makes it easy to see what a given CSDB object "depends" on.

2 Usage

```
s1kd-refls [-qcaNh?] <objects>...
```

3 Options

- | | |
|-------|--|
| -a | List all references, not attempting to match them to an actual filename. |
| -c | List references in the <code>content</code> section of a CSDB object only. |
| -h -? | Show help/usage message. |

- N Assume filenames of referenced CSDB objects omit the issue info, i.e. they were created with the -N option to the s1kd-new* tools.
- q Quiet mode. Errors are not printed.

s1kd-validate

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1
3.1 Multi-spec directory with -d option.....	2
3.2 XML catalogs vs. -d option.....	2

List of tables

1	References.....	1
---	-----------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 General

The **s1kd-validate** tool validates an S1000D data module, checking whether it is a valid XML file and if it is valid against its own S1000D schema.

2 Usage

```
s1kd-validate [-d <dir>] [-X <URI>] [-vqD] [<datamodule>...]
```

3 Options

-d <dir>

Search for schemas in <dir>. Normally, the URI of the schema is used to fetch it locally or over a network, but this option will force searching to be performed only in the specified directory.

	This can also be accomplished through the use of XML catalogs.
-f	List invalid files.
-v -q -D	Set the verbosity of the output, verbose, quiet, and debug. Verbose will explicitly indicate success, rather than simply not displaying any errors. Quiet will not output anything.
-X <URI>	Exclude an XML namespace from the validation. Elements in the namespace specified by <URI> are ignored.
<datamodule>...	Any number of data modules to validate. If none are specified, input is read from stdin.

3.1 Multi-spec directory with -d option

The -d option can point either to a directory containing the XSD schema files for a single S1000D spec (i.e. the last part of the schema URI), or to a directory containing schemas for multiple specs. The latter must follow a particular format for the tool to locate the appropriate schemas for a given spec:

```
schemas/      <-- The directory passed to -d
  S1000D_4-1/
    xml_schema_flat/
      [4.1 XSD files...]
  S1000D_4-2/
    xml_schema_flat/
      [4.2 XSD files...]
```

3.2 XML catalogs vs. -d option

XML catalogs provide a more standard method of redirecting public, network-based resources to local copies. As part of using libxml2, there are several locations and environment variables from which this tool will load catalogs.

Below is an example of a catalog file which maps the S1000D schemas to a local directory:

```
<catalog xmlns="urn:oasis:names:tc:entity:xmlns:xml:catalog">
<rewriteURI uriStartString="http://www.s1000d.org/"
rewritePrefix="/usr/share/slkd/schemas/" />
</catalog>
```

This can be placed in a catalog file automatically loaded by libxml2 (e.g., /etc/xml/catalog) or saved to a file which is then specified in an environment variable used by libxml2 (e.g., XML_CATALOG_FILES) to remove the need to use the -d option.

s1kd-acronyms

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1

List of tables

1 References.....	1
------------------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 **General**

The **s1kd-acronyms** tool generates a list of unique acronyms used in S1000D data modules. It can also mark up acronyms in data modules automatically based on an existing list.

2 **Usage**

```
s1kd-acronyms [-pxdtiIh?] [-n <#>] [-T <types>]
               [-m <acronyms>] [-o <file>]
               [<datamodules>]
```

3 **Options**

-d	Format XML output as an S1000D <definitionList>.
-h -?	Show help/usage message.
-i -I	Markup acronyms in interactive mode. If the specified acronyms list contains multiple definitions for a given

acronym term, the tool will prompt the user with the context in which the acronym is used and present a list of the definitions for them to choose from.

When not in interactive mode, the first definition found will be used.

The -l option prompts for all acronyms, not just those with multiple definitions. This can be useful if some occurrences of the acronym term should be ignored.

-m <acronyms>

Instead of listing acronyms, automatically markup acronyms given in the <acronyms> XML file in the specified data modules. Occurrences of the acronym term will be replaced in text with the `acronym` element in the list.

-n <#>

Minimum number of spaces after the term in pretty-printed text output.

-o <file>

Output to <file> instead of stdout. When used with the -m option, output to <file> instead of overwriting the existing file.

-p

Pretty print text/XML output.

-T <types>

Only search for acronyms with an attribute `acronymType` whose value is contained within the string <types>.

-t

Format XML output as an S1000D <table>.

-x

Use XML output instead of plain text.

<datamodules>

Data modules to find acronyms in.

s1kd-flatpm

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1

List of tables

1 References.....	1
------------------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 **General**

The **s1kd-flatpm** tool combines a publication module and the data modules it references in to a single file for use with a publishing system.

Data modules are searched for in the current directory using the data module code, language and/or issue info provided in each reference.

2 **Usage**

```
s1kd-flatpm [-Nxp] <PM> [<DM>...]
```

3 **Options**

-h -?

Show help/usage message.

-N

Assume that the files representing the referenced data modules do not include the issue info in their filenames, i.e. they were created using the -N option of the s1kd-new* tools.

-p	Instead of the "flat" PM format, use a "publication" XML format, where the root element <code>publication</code> contains <code>XInclude</code> references to the publication module and the referenced data modules.
-x	Use <code>XInclude</code> rather than copying each data module's contents directly inside the publication module. DTD entities in data modules will only be carried over to the final publication when using this option, otherwise they do not carry over when copying the data module.
<DM>...	When using the -p option, the filenames to include can be specified manually as additional arguments instead of searching for them in the current directory. When not using the -p option, additional arguments are ignored.
<PM>	The publication module to flatten.

s1kd-instance

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	2
3 Options.....	2
3.1 -a vs -A.....	4
3.2 Identifying source module of an instance.....	5
3.3 Instance data module/publication module code (-c) vs extension (-e).....	5
3.4 Filtering for multiple values of a single property.....	5
3.5 Resolving CIR dependencies with a custom XSLT script (-r).....	7

List of tables

1	References.....	1
---	-----------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 General

The **s1kd-instance** tool produces an "instance" of an S1000D data module or publication module, derived from a "master" (or "source") module. The tool supports multiple methods of instantiating a module:

- Filtering on user-supplied applicability definitions, so that non-applicable elements and (optionally) unused applicability statements are removed in the instance. The definitions can be supplied directly or read from a PCT (Product Cross-reference Table).
- Using a CIR (Common Information Repository) to produce a standalone instance from a CIR-dependent master.
- Changing various pieces of metadata in the instance.

Any combination of these methods can be used when producing an instance.

2 Usage

```
slkd-instance [-s <src>] [-e <ext>|-E] [-c <dmc>]
              [-l <lang>] [-n <iss>] [-I <date>]
              [-u <sec>] [-o <file>|-O <dir>] [-f]
              [-t <techName>] [-i <infoName>] [-a|-A]
              [-Y <text>] [-C <comment>]
              [-R <CIR> ...] [-r <XSL>] [-x <CIR>]
              [-S] [-N] [-P <PCT> -p <id>] [-L]
              [<applic>...]
```

3 Options

-A	Remove unused applicability annotations and simplify/remove unused applicability statements.
-a	Remove unused applicability annotations but not statements.
-C <comment>	Add an XML comment to the top of the instance. Useful as another way of identifying a module as an instance aside from the source address or extended code, or giving additional information about a particular instance.
-c <dmc>	Specify a new data module code (DMC) or publication module code (PMC) for the instance.
-E	Remove the extension from an instance produced from an already extended module.
-e <ext>	Specify an extension on the data module code (DME) or publication module code (PME) for the instance.
-f	Overwrite existing file with same name as the filename generated automatically with -O, if it exists.
-I <date>	Set the issue date of the instance. By default, the issue date is taken from the source.
-i <infoName>	Give the data module instance a different infoName.
-L	Source (-s or stdin) is a list of module filenames to create instances of, rather than a single module.
-l <lang>	Set the language and country of the instance. For example, to create an instance for US English, lang would be "en-US".
-N	Omit issue/inwork numbers from automatically generated filenames.
-n <iss>	Set the issue and inwork numbers of the instance. By default, the issue and inwork number are taken from the source.
-O <dir>	Output instance(s) in dir, automatically naming them based on:

- the extension specified with -e
- the code specified with -c
- The issue info specified with -n
- the language and country specified with -L

If any of the above are not specified, the information is copied from the source module.

-o <file>

Output instance to file instead of stdout.

-P <PCT>

PCT file to read product definitions from (-p).

-p <id>

Product ID of the product to read applicability definitions from, using the specified PCT data module (-P).

-R <CIR> ...

Use a CIR to resolve external dependencies in the master data module, making the instance data module standalone. Additional CIRs can be used by specifying the -R option multiple times.

The following CIRs have some built-in support:

- Access points
- Applicability
- Cautions
- Circuit breakers
- Controls/indicators
- Enterprises
- Functional items
- Parts
- Supplies
- Tools
- Warnings
- Zones

The methods of resolving the dependencies for a CIR can be changed by specifying a custom XSLT script with the -r option. The built-in XSLT used for the above CIR data modules can be dumped with the -x option.

-r <XSL>

Use a custom XSLT script to resolve CIR dependencies for the last specified CIR.

-S

Do not include <sourceDmIdent>/<sourcePmIdent>/<repositorySourceDmIdent> in the instance.

-s <src>

The source module (default is to read from stdin).

-t <techName>	Give the instance a different techName/pmTitle.
-u <sec>	Set the security classification of the instance. An instance may have a lower security classification than the source if classified information is removed for a particular customer.
-v	When -O is used, print the automatically generated file name of the instance.
-w	Check the applicability of the whole module against the user-defined applicability. If the whole module is not applicable, then no instance is created.
-x <CIR>	<p>Dumps the built-in XSLT used to resolve dependencies for <CIR> CIR type to stdout. This can be used as a starting point for a custom XSLT script to be specified with the -r option.</p> <p>The following types currently have built-in XSLT and can therefore be used as values for <CIR>:</p> <ul style="list-style-type: none">- accessPointRepository- applicRepository- cautionRepository- circuitBreakerRepository- controllIndicatorRepository- enterpriseRepository- functionalItemRepository- partRepository- supplyRepository- toolRepository- warningRepository- zoneRepository
-Y <text>	Set the applicability for the whole module using the user-defined applicability values, using text as the new display text.
<applic>...	Any number of applicability definitions in the form of: <ident>:<type>=<value>

3.1

-a vs -A

The -a option will remove applicability annotations (applicRefId) from elements which are deemed to be unambiguously valid (their validity does not rely on applicability values left undefined by the user). The applicability statements themselves however will be untouched.

The -A option will do the above, but will also attempt to simplify unused parts of applicability statements or remove unused applicability statements entirely. It simplifies a statement by

removing <assert> elements determined to be either unambiguously valid or invalid given the user-defined values, and removing unneeded <evaluate> elements when they contain only one remaining <assert>.

Note

The -A option may change the **meaning** of certain applicability statements without changing the **display text**. Display text is always left untouched, so using this option may cause display text to be technically incorrect.

3.2 Identifying source module of an instance

The resulting data module instance will contain the element <sourceDmIdent>, which will contain the identification elements of the data module specified with the -s option. Publication module instances will contain the element <sourcePmIdent> instead.

Additionally, the data module instance will contain an element <repositorySourceDmIdent> for each CIR specified with the -R option.

If the -S option is used, neither the <sourceDmIdent>/<sourcePmIdent> elements or <repositorySourceDmIdent> elements are added. This can be useful when this tool is not used to make an "instance" per se, but more generally to make a module based on an existing module.

3.3 Instance data module/publication module code (-c) vs extension (-e)

When creating a data module or publication module instance, the instance should have the same data module/publication module code as the master, with an added extension code, the DME/PME. However, in cases where a vendor does not support this extension or possibly when this tool is used to create "instances" which will from that point on be maintained as normal standalone data modules/publication modules, it may be desirable to change the data module/publication module code instead. These two options can be used together as well to give an instance a new DMC/PMC as well an extension.

3.4 Filtering for multiple values of a single property

Though not usually the case, it is possible to create an instance which is filtered on multiple values of the same applicability property. Given the following:

```
<referencedApplicGroup>
<applic id="apA">
<assert applicPropertyIdent="attr"
applicPropertyType="prodattr"
applicPropertyValues="A" />
</applic>
<applic id="apB">
<assert applicPropertyIdent="attr"
applicPropertyType="prodattr"
applicPropertyValues="B" />
</applic>
<applic id="apC">
<assert applicPropertyIdent="attr"
applicPropertyType="prodattr"
```



```
applicPropertyValues="C"/>
</applic>
</referencedApplicGroup>
<!-- ... -->
<para applicRefId="apA">Applies to A</para>
<para applicRefId="apB">Applies to B</para>
<para applicRefId="apC">Applies to C</para>
```

filtering can be applied such that the instance will be applicable to both A and C, but not B. This is done by specifying a property multiple times in the applicability definition arguments. For example:

```
$ slkd-instance -A -Y "A or C" ... attr:prodattr=A attr:prodattr=C
```

This would produce the following in the instance:

```
<dmStatus>
  <!-- ... -->
  <applic>
    <displayText>
      <simplePara>A or C</simplePara>
    </displayText>
    <evaluate andOr="or">
      <assert applicPropertyIdent="attr"
        applicPropertyType="prodattr"
        applicPropertyValues="A"/>
      <assert applicPropertyIdent="attr"
        applicPropertyType="prodattr"
        applicPropertyValues="C"/>
    </evaluate>
  </applic>
  <!-- ... -->
</dmStatus>
<!-- ... -->
<referencedApplicGroup>
  <applic id="apA">
    <assert applicPropertyIdent="attr"
      applicPropertyType="prodattr"
      applicPropertyValues="A"/>
  </applic>
  <applic id="apC">
    <assert applicPropertyIdent="attr"
      applicPropertyType="prodattr"
      applicPropertyValues="C"/>
  </applic>
</referencedApplicGroup>
<!-- ... -->
<para applicRefId="apA">Applies to A</para>
<para applicRefId="apC">Applies to C</para>
```

3.5 Resolving CIR dependencies with a custom XSLT script (-r)

A CIR contains more information about an item than can be captured in a data module's reference to it. If this additional information is required, there are two methods to include it:

- Distribute the CIR with the data module so the extra information can be linked to
- "Flatten" the information to fit in the data module's schema.

A custom XSLT script can be supplied with the -r option, which is then used to resolve the CIR dependencies of the last CIR specified with -R. For example:

```
<xsl:stylesheet
xmlns:xsl="http://www.w3.org/1999/XSL/Transform"
version="1.0">
<xsl:template match="functionalItemRef">
<xsl:variable name="fin" select="@functionalItemNumber"/>
<xsl:variable name="spec" select="$cir//functionalItemSpec[
functionalItemId/@functionalItemNumber = $fin]"/>
<xsl:value-of select="$spec/name"/>
</xsl:template>
</xsl:stylesheet>
```

This script would resolve a `functionalItemRef` by "flattening" it to the value of the `name` element obtained from the CIR.

The example CIR would contain a specification like:

```
<functionalItemSpec>
<functionalItemId functionalItemNumber="ABC"
functionalItemType="fit01"/>
<name>Hydraulic pump</name>
<functionalItemAlts>
<functionalItem/>
</functionalItemAlts>
</functionalItemSpec>
```

The source data module would contain a reference:

```
<para>
The
<functionalItemRef functionalItemNumber="ABC"/>
is an item in the system.
</para>
```

The command would resemble:

```
$ slkd-instance -s <src> -R <CIR> -r <custom XSLT>
```

And the resulting XML would be:

```
<para>The Hydraulic pump is an item in the system.</para>
```

The source data module and CIR are combined in to a single XML document which is used as the input to the XSLT script. The root element `mux` contains two `dmodule` elements. The first is the source data module, and the second is the CIR data module specified with the corresponding `-R` option. The CIR data module is first filtered on the defined applicability.

An "identity" template is automatically inserted in to the custom XSLT script, equivalent to the following:

```
<xsl:template match="@*|node()">
<xsl:copy>
<xsl:apply-templates select="@*|node()" />
</xsl:copy>
</xsl:template>
```

This means any elements or attributes which are not matched with a more specific template in the custom XSLT script are automatically copied.

The set of built-in XSLT scripts used to resolve dependencies can be dumped using the `-x` option.

s1kd-neutralize

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1
4 Example.....	2

List of tables

1 References.....	1
------------------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 **General**

Generates neutral metadata for the specified data modules. This includes:

- XLink attributes for references, using the S1000D URN scheme.
- RDF and Dublin Core metadata.

2 **Usage**

```
s1kd-neutralize [-o <file>] [-fh?] [<data module> ...]
```

3 **Options**

- | | |
|-----------|---|
| -f | Overwrite specified data module(s) automatically. |
| -h -? | Show usage message. |
| -o <file> | Output neutralized data module XML to <file> instead of stdout. |

4 Example

```
$ DMOD=DMC-XLINKTEST-A-00-00-00-00A-040A-D_000-01_EN-CA.XML
$ xmllint --xpath "//description/dmRef" $DMOD
<dmRef>
<dmRefIdent>
<dmCode modelIdentCode="XLINKTEST" systemDiffCode="A"
systemCode="00" subSystemCode="0" subSubSystemCode="0" assyCode="01"
disassyCode="00" disassyCodeVariant="A" infoCode="040"
infoCodeVariant="A" itemLocationCode="D"/>
</dmRefIdent>
<dmRefAddressItems>
<dmTitle>
<techName>XLink test</techName>
<infoName>Referenced data module</infoName>
</dmTitle>
</dmRefAddressItems>
</dmRef>

$ slkd-neutralize $DMOD | xmllint --xpath "//description/dmRef" -
<dmRef xlink:type="simple"
xlink:href="URN:S1000D:DMC-XLINKTEST-A-00-00-01-00A-040A-D"
xlink:title="XLink test - Referenced data module">
[...]
</dmRef>
```

s1kd-syncrefs

Description

Table of contents

Page

Description.....	1
References.....	1
Description.....	1
1 General.....	1
2 Usage.....	1
3 Options.....	1

List of tables

1 References.....	1
------------------------	---

References

Table 1 References

Data module/Technical publication	Title
None	

Description

1 **General**

The **s1kd-syncrefs** tool copies all external references (dmRef, pmRef, externalPubRef) within the content of a data module and uses them to generate the <refs> element. Each unique reference is copied, sorted, and placed in to the <refs> element. If a <refs> element already exists, it is overwritten.

2 **Usage**

```
s1kd-syncrefs [-df] [-o <out>] [<data module>...]
```

3 **Options**

-d	Delete the <refs> element.
-f	Overwrite the data modules automatically.
-o <out>	The resulting XML is written to <out> instead of stdout.
<data module>...	The data module(s) to synchronize references in. Default is to read from stdin.