

User Guide for the Core2Smv Project¹

Last Updated: 15/06/2006

Table of Contents

1. Introduction	2
2. Relevant Files	3
3. Running the Core2Smv Translation Program (Linux Version)	4
4. Running the Flat Program (Linux Version)	5
5. Running the Translation Program (Other PC Versions)	6
6. Configuration File & Default Values	7
7. Limitations of the Translation	8
8. Trouble-Shooting	9
9. Contact Information	9

¹ It is recommended to read this document electronically, enabling usage of cross references and hyperlinks

1. Introduction

Core2Smv is a software package used for translating a core program to an equivalent smv program.

The translation was developed at VeriTech, SSDL laboratory, Faculty of Computer Science, Technion, under supervision of Prof. Shmuel Katz and Mr. Shahar Dag.

The VeriTech homepage contains information regarding core (or CDL – core definition language) and smv languages: description, grammar, examples, etc.

VeriTech homepage:

www.cs.technion.ac.il/~ssdl/veritech/

Detailed description of the translation (ideas, algorithms, data structures, features, etc.) exists as a separate document ([see Core2Smv.doc, below](#)).

Basically, the translation is done in three stages, each one executed by a specific application:

1. CoreToXml – an application receiving as input a core program in textual form and producing a XML version of this core program.
2. Core2Smv – the translation program (from core to smv).
3. xml2txt – an application receiving as input a smv program in XML form and producing a text version of the smv program.

These applications are independent and are linked by a script file.

Apart from the translation program, a 'Flat' application is also supplied. This application "flattens" a core program – a process of transforming a core program to an equivalent one containing only one module.

2. Relevant Files

The Core2Smv project includes the following files and directories, currently located under the directory

~veritech/Veritech/Core2Smv 2.0/

- readme.txt:
 - Command line usage for running the Core2Smv translation program.
 - Command line usage for running the Flat program.
- Executable jar files:
 - Core2Smv.jar – executable for running the translation program.
 - Flat.jar – executable for flattening the entire core program.
- Script files:
 - Core2Smv
 - Flat
 - DoAll – script for running all the examples.
 - UnDoAll – script for deleting all the products of DoAll.
- Makefile - makefile for compiling sources and creating executables.
- Manifest files:
 - Manifest.txt - marks the 'main' class to be run by Core2Smv.jar
 - Manifest_flat - marks the 'main' class to be run by Flat.jar
- config.txt – configuration file.
- db_alon.xml – database file for web-generation of the Core2Smv project.
- src - a subdirectory containing the java source files.
- docs - a subdirectory containing documentation
 - docs/Core2Smv.doc – detailed documentation.
 - docs/User Guide.doc – this file.
 - docs/variable_assignments.doc – intermediate article discussing variable assignments and state-to-state transitions.
 - docs/partial_synchronization.doc – intermediate article discussing elimination of partial synchronizations in a core program.
 - docs/comments.doc – a few comments on the input files and external tools.
 - docs/javadocs – a subdirectory with 'JavaDoc' documentation
- examples - a subdirectory containing tested examples
- eclipse_prj – a subdirectory containing eclipse project settings.

Other files used by the translation are dtd and xsl files (compiled at Veritech); their url's are:

www.cs.technion.ac.il/~ssdl/veritech/xml/dtd/cdl.dtd
www.cs.technion.ac.il/~ssdl/veritech/xml/dtd/smv.dtd
www.cs.technion.ac.il/~ssdl/veritech/xml/dtd/add_info.dtd
www.cs.technion.ac.il/~ssdl/veritech/xml/xsl/xml2cdl.xsl
www.cs.technion.ac.il/~ssdl/veritech/xml/xsl/xml2smv.xsl
www.cs.technion.ac.il/~ssdl/veritech/xml/xsl/xml2addinfo.xsl

3. Running the Core2Smv Translation Program (Linux Version)

The following files are necessary for running the Core2Smv program:

- Core2Smv.jar
- Core2Smv (the script file)
- Manifest.txt
- config.txt

The command line usage (as seen also in 'readme.txt') is:

```
Core2Smv [-xml] <core filename> [-coreDS] [-smvDS] [-ext] [-flat]
        [-min <min int>] [-max <max int>]
        [-warnArray <max array size>]
        [-warnProgram <max array elements in program>]
```

This command invokes the 'Core2Smv' script file and passes the arguments to the script.

The first argument is optional and serves as a flag for the input file type.

- -xml: Use this flag to indicate that the input file contains the core program already in its XML form, and does not need to be parsed. This flag, if used, must precede all other arguments.

The next argument is mandatory, and must be included in the command line.

- core filename: File name of the inputted core program. This can be either a text file or a XML file.
 - In case of a text file, do not use the '-xml' switch. The script will invoke the CoreToXml program which will use this text file and create a XML file of the core program. The newly created XML file will be located in the same directory as the inputted text file; its name will be composed of the text file's name, suffixed with '_xml.xml'. For example, if the inputted text file is named 'example.cdl', the xml file will be named 'example.cdl_xml.xml'.
 - In case of a XML file – use the '-xml' switch.

All other arguments are optional: They may appear in any order, but if used – they must appear after the '-xml' flag (if used) and after the core file name.

- -coreDS: A flag indicating textual downloading of the core data structure. If this flag is used, and the configuration file specifies printing of data structures, the core data structure will be downloaded. ([See also 6. Configuration File & Default Values.](#))
- -smvDS: A flag indicating textual downloading of the smv data structure. If this flag is used, and the configuration file specifies printing of data structures, the smv data structure will be downloaded. ([See also 6. Configuration File & Default Values.](#))
- -ext: A flag indicating inclusion of "extended" ("additional" + "regular") information in the additional information file. If this flag is not used, only additional information is included.

In this context, "additional" information is the set of all non-trivial links between core objects and smv objects. "regular" information is the set of all trivial links.

"trivial" means that by looking at a target object one can deduce the source object without having the source program at hand

- -flat: A flag enforcing flattening of the whole core program prior to its translation. If this flag is not used, the flattening decision is made by the translation program (avoiding flattening as much as possible).
- -min: Use this switch followed by an integer representing the desired minimum integer value in the smv program. This will override the default MININT value. (See also [6. Configuration File & Default Values](#) and [7.1 MIN_INT / MAX_INT limits](#).)
- -max: Use this switch followed by an integer representing the desired maximum integer value in the smv program. This will override the default MAXINT value. (See also [6. Configuration File & Default Values](#) and [7.1 MIN_INT / MAX_INT limits](#).)
- -warnArray: Use this switch followed by an integer representing the maximum number of elements per array allowed in the smv program. This will override the default value. (See also [6. Configuration File & Default Values](#) and [array elements per variable limitation in 7.2](#).)
- -warnProgram: Use this switch followed by an integer representing the maximum number of array elements allowed in the whole smv program. This will override the default value. (See also [6. Configuration File & Default Values](#) and [array elements per program limitation in 7.2](#).)

4. Running the Flat Program (Linux Version)

The following files are necessary for running the Flat program:

- Flat.jar
- Flat (the script file)
- Manifest_flat.txt
- config.txt

The command line usage (as seen also in 'readme.txt') is:

```
Flat [-xml] <core filename> [-coreDS]
```

This command invokes the 'Flat' script file and passes the arguments to the script. The command line arguments are exactly the same as the first three arguments passed to the Core2Smv program, and treated in the same manner.

5. Running the Translation Program (Other PC Versions)

It is also possible to run the translation program in any standard java environment. I used 'Eclipse' (Windows version) at the development and testing stages (Eclipse version 3.1.2. compiler compliance 1.5).

However, CoreToXml and xml2txt are Linux-platform applications and can not be trusted to run on Windows. Since only the Core2Smv application can be run,

- The input core program must be in XML format.
- In order to produce a text version of the smv program, use the '-smvDS' flag.

The java source files can be copied from the 'src' subdirectory.

Create a java project and compile the files.

The Eclipse project settings are included in the 'eclipse_prj' subdirectory.

Use the same program arguments listed above, in the same order, except the '-xml' flag. Do not include this flag as an argument. As mentioned above, this flag is intended for avoiding the call to the CoreToXml utility in the Linux script file before calling the java program, and is not an argument of the java program itself.

The 'Flat' program can also be run on Windows. As with the Core2Smv application, the input must be a XML file, and the '-xml' flag should not be used.

On a Windows platform, jar files should be invoked (as opposed to invoking scripts on Linux). The proper command lines are:

```
java -jar Core2Smv.jar <core xml filename> [-coreDS] [-smvDS]
    [-ext] [-flat] [-min <min int>] [-max <max int>]
    [-warnArray <max array size>]
    [-warnProgram <max array elements in program>]

java -jar Flat.jar <core xml filename> [-coreDS]
```

6. Configuration File & Default Values

The configuration file (config.txt) contains several parameters used by the translation program.

When running the translation program, the configuration file is looked for, and the contained parameters are extracted. If it is not found, a default configuration file is created. If still not found, or if it is syntactically incorrect, the translation is aborted and an error message is printed.

The form of the default configuration file is:

```
MIN_INT = -100
MAX_INT = 100
MAX_ARRAY_ELEMENTS_PER_VARIABLE = 100
MAX_ARRAY_ELEMENTS_PER_PROGRAM = 300
SMV_DTD_FILE = "http://www.cs.technion.ac.il/~ssdl/
                veritech/xml/dtd/smv.dtd"
ADD_INFO_DTD_FILE = "http://www.cs.technion.ac.il/~ssdl/
                    veritech/xml/dtd/add_info.dtd"
INCLUDE_EXTENDED_INFO = false
SHOULD_FLAT = false
PRINT_DATA_STRUCTURES = false
```

The default configuration file holds parameter values which are default values, also hard-coded in the translation program. Overriding these hard-coded values is done by modifying the configuration file.

The following parameters can be further overridden by command line flags ([see also command line flags in 3 – Running the Core2Smv Translation Program](#)):

```
MIN_INT
MAX_INT
MAX_ARRAY_ELEMENTS_PER_VARIABLE
MAX_ARRAY_ELEMENTS_PER_PROGRAM
INCLUDE_EXTENDED_INFO
SHOULD_FLAT
```

If 'PRINT_DATA_STRUCTURES' is true, data structures are downloaded only upon using the '-coreDS' or '-smvDS' command line flags.

If 'PRINT_DATA_STRUCTURES' is false and the command line flags '-coreDS' or '-smvDS' are used, data structures will not be downloaded, and a warning message will be printed.

The dtd file names can not be overridden by command line flags.

7. Limitations of the Translation

1. MIN_INT and MAX_INT:

- These parameters should be in the java integer range.
- Smv does not have an integer type. Core integer-typed variables are translated to smv range-typed variables.
- The translation does not allow out-of-bounds assignments to range-typed variables. Therefore, some integer assignments (in the core program) will not be executed in the smv program.

Variable assignments represent transitions between successive states. If a variable is assigned with a value which may possibly be out of range, a condition restricting the transition is specified in the smv program. For example, if the core program contains the assignment $x' = t$ (where x is integer typed), the smv program will specify that a transition to the next state is possible only if $\text{MIN_INT} \leq t \leq \text{MAX_INT}$.

(See also sections [6.3.4 – Out of Bound Check](#) and [6.3.7 – Avoiding Incorrect Assignments in Core2Smv.doc.](#))

2. Array element overflow:

- Limits are set on the number of array elements in the smv program. The reasoning behind this decision is to avoid a vast number of code lines in the output smv program (see also section [6.3.5. – Array Element Assignment in Core2Smv.doc.](#)).
- A limit is set on the maximum number of array elements per array-typed variable. If this limit is exceeded, the translation is aborted and a message is printed.
- A limit is set on the maximum number of array elements per program. If this limit is exceeded, the program will print a warning message, and continue the translation without further expansion of array elements, thus resulting in an incorrect smv program. Array elements are expanded until the limit is reached. We considered this behavior better than no expansion at all.

3. Validity of the input core program:

- The translation does not always check whether the inputted core program is grammatically correct. Feeding invalid core programs may result in unexpected behavior.
- Some cases of incorrect syntax or grammar are handled. If encountered, the translation is aborted and an error message is printed. One such case is a core program having no 'SYSTEM' module.
- Testing has shown that several types of "small mistakes" in core are translated to "equivalent small mistakes" in smv:
 - If a core module has two arguments with the same name, the corresponding smv module will also have two arguments with the same name.
 - If a core module contains two variables with the same name, the corresponding smv module will contain only one of them.
 - If the core program contains two modules with the same name, the smv program will contain only one of them.
 - Some cases of undeclared identifiers are not noticed by the translation program. For example, if the core program contains the expression ' $k < 5$ ', and k is not declared, the expression will be translated without an error or a

warning message. However, upon an assignment to an undeclared variable, a warning message will be printed.

8. Trouble-Shooting

- 'Core2Smv command not found':
Add the full path of the directory containing the project files to the 'path' environment variable, using a 'set path' command. For example, to add the current directory to the path variable, write:

```
set path = (. $path)
```

This command depends on the shell used.
- 'CoreToXml command not found':
The CoreToXml application is invoked by the script, which contains the following 'set path' command to identify the path to this application:

```
set path = ($path /home/veritech/VeriTech/CoreParser/bin)
```

Its location may have been changed.
Insert the proper path in the script line.
- 'cdl.dtd file not found':
The CoreToXml application specifies the path to the cdl.dtd file in one of the first lines in the XML file it produces. The path may have been changed, or the dtd file may not exist in the specified directory. Do one of the following:
 - Place a copy the cdl.dtd file in the directory in which the input core program resides.
 - Change the path to the cdl.dtd file in the XML file produced by the CoreToXml application.
- 'xml2smv.xsl file not found':
The xml2txt application expects a full path to the xml2smv.xsl file. The path may have been changed, or the xsl file may not exist in the specified directory. Do one of the following:
 - Place a copy the xml2smv.xsl file in the directory in which the input core program resides.
 - Specify the full path to the xml2smv.xsl file in the script file (in the line invoking the xml2txt application).

9. Contact Information

Alon Zvirin salz@csd.technion.ac.il 052-4251526, 04-8624341