NAME

svl – compilation and execution of SVL scripts

SYNOPSIS

svl *options* [*file*[**.svl**] [*script-parameters*]]
svl -script *options* [*file1*[**.svl**]] [-output *file2*]
svl -expand [-case] [-indent *n*] [*file1*[**.svl**] **] [-output** *file2* **]**
svl -clean [*file***[.svl]]**
svl -sweep [*file***[.svl]]**
svl -help
svl -version
where the following *options* **are available:**
-case, -debug, -ignore, -sh *sh-options***, -silent, -v** *variable***=***value***.**

DESCRIPTION

In the simplest form (first line of the synopsis), taking as input *file***.svl**, which contains a verification script in SVL (Script Verification Language), **svl** produces actions invoking CADP tools. See **svl-lang**(LOCAL) for a description of the SVL language.

svl runs a given script through the following steps:

- The program is compiled to a Bourne shell-script that contains a sequence of invocations to the appropriate CADP tools.
- The generated script is executed with optional *script-parameters*, which can be used within the SVL script as Bourne shell parameters $1, $2, ...
- Finally, the generated script is erased.

During script execution, several temporary intermediate files may be created, and removed as soon as possible in order to minimize disk space consumption. **svl** also maintains a log file named *file***.log** that contains a full diagnostic of the script execution.

COMMANDS

svl can also be used for alternative actions, using the commands described below:

-script Stop after generation of the Bourne shell script. If the parameter *file2* is specified, then write the generated script to this file. Otherwise display it on the standard output.

-expand
Stop after the expansion phase i.e., after propagating meta-operators inside expressions. If the parameter *file2* is specified, then write the expanded SVL program to this file. Otherwise display the expanded program on the standard output.

-clean Remove the generated shell script and all files created during the latest execution of *file***.svl**. Note that dynamic libraries of BCG files are erased at the same time as the BCG files themselves, and that files created by user written shell commands are not removed. For this command to be effective, both *file***.log** and *file***.svl** must exist in the current directory.

-sweep Remove temporary files reminiscent of a run in **-debug** mode (see options below). Note that files created by user written shell commands are not removed. The log file and the output files are not removed either.

-help Display the available options of **svl**.

-version
Display the current version number of **svl**.

Note that *file***.svl** (or *file1***.svl**) can be omitted if there exists a unique file with extension **.svl** in the current directory, in which case this file is considered as the input.

OPTIONS

The following options are currently available:

-case The default behaviour of **svl** is to not distinguish lowercase and uppercase characters in identifiers that are not enclosed in double quotes (all characters are turned to uppercase). The **-case** option forces the distinction.

-debug Stop execution has soon as an invoked tool returns a non-zero exit status. Do not erase intermediate files during execution of the generated script, and do not erase the script file after execution. Use command **-sweep** or **-clean** to erase these files afterwards.

-ignore Do not stop execution of the script after an error is issued, continue execution until all instructions of the script are executed (possibly with errors).

-indent *n*
Set indentation to *n* blank characters for expanded SVL script display (command **-expand**). Default value for *n* is 3.

-sh *sh-options*
Pass *sh-options* to the Bourne shell interpreter of the generated script. Note that *sh-options* must count as a single argument, hence it may be necessary to use quotes to group several options.

-silent Do not comment actions of the script during execution. This is not a default option.

-v *variable*=*value*
Add the shell variable definition ''*variable*=*value*'' to the generated shell script. This variable definition will occur just after the inclusion of the **standard** file (see Section ENVIRONMENT VARIABLES below), so that the initializations done in **standard** cannot overwrite the definition of *variable*.
Option **-v** can be used multiple times on the same command line. The corresponding variable definitions will then occur in the generated shell script in the same order as they occur on the command line.
Note that if *value* contains shell-interpreted characters that should be left unchanged by the command-line interpreter, then it should either be written between single quotes, or every shell-interpreted character should be escaped by a backslash (`\') character.
See Section LOCAL SHELL VARIABLES in the **svl-lang**(LOCAL) manual page for a complete list of the shell variables used by SVL scripts.

ENVIRONMENT VARIABLES

The following environment variables are used:

$CADP Needed. This variable contains the path of directory where CADP is installed.

$CADP/com
This directory should be put in the **$PATH** variable.

$SVL Optional. The first action of the generated script is to include the file **$CADP/src/svl/standard**, containing a list of predefined shell functions and variables. However, if the environment variable **SVL** is defined, the included file is **$SVL/src/svl/standard**. Moreover, the kernel program **`svl_kernel`** will be searched in **$SVL/bin.‘arch‘** instead of **$CADP/bin.‘arch‘**.

EXIT STATUS

When the source is erroneous, error messages are issued. Exit status is 0 if everything is alright, 1 otherwise.

AUTHORS

svl version 2.0 and later: Hubert Garavel, Frederic Lang, Marc Herbert.

svl versions 1.0 to 1.6 (kept for internal use and never distributed officially): Mark Jorgensen, Christophe Discours, Hubert Garavel.

The authors owe thanks to Radu Mateescu and Charles Pecheur for their feedback about **svl** 1.∗ and to Laurent Mounier for his useful advices.

FILES

The following files and extensions are handled by **svl**, either as inputs, outputs, or as temporary intermediate files.

Extension	Description	In	Out	Tmp
.lnt	LNT program	X		
.lotos .lot	LOTOS program	X		
.lts	FSP program	X		
.aut	LTS in AUT format | X	X	X	
.bcg	LTS in BCG format	X	X	X
.fc2	LTS in FC2 format	X	X	X
.seq	LTS in sequence format	X	X	
.exp	network of LTSs	X	X	X
.hide .hid	labels to hide	X		X
.cut	labels to cut	X		X
.rename .ren	labels to rename	X		X
.sync	labels to synchronize	X		X
.mcl	temporal logic formula	X		
.log	log of execution		X	

Note: **svl** generates temporary files in the current directory. To avoid clashes with existing files, file names generated by **svl** are prefixed with a string of the form **`svl`***xxx***`_`**, where *xxx* is a 3 digits number. Hence it is recommended to avoid naming user files this way.

BIBLIOGRAPHY

[GL01] Hubert Garavel and Frederic Lang. SVL: a Scripting Language for Compositional Verification. In Myungchul Kim, Byoungmoon Chin, Sungwon Kang, and Danhyung Lee (editors), Proceedings of the 21st International Conference on Formal Techniques for Networked and Distributed Systems FORTE'2001 (Cheju Island, Korea), IFIP Conference Proceedings volume 197, pages 377-394, Kluwer, August 2001. Available from http://cadp.inria.fr/publications/Garavel-Lang-01.html

[GLM15] Hubert Garavel, Frederic Lang, and Radu Mateescu. Compositional Verification of Asynchronous Concurrent Systems using CADP. Acta Informatica, Special Issue on Combining Compositionality and Concurrency: Part 2, 52(4-5):337-392, 2015. Available from http://cadp.inria.fr/publications/Garavel-Lang-Mateescu-15.html

[KM97] Jean-Pierre Krimm and Laurent Mounier. Compositional State Space Generation from LOTOS Programs. In Ed Brinksma (editor), Proceedings of TACAS'97 Tools and Algorithms for the Construction and Analysis of Systems (University of Twente, Enschede, The Netherlands), Lecture Notes in Computer Science volume 1217, Springer, April 1997. Available from http://cadp.inria.fr/publications/Krimm-Mounier-97.html

[Lan02] Frederic Lang. Compositional Verification using SVL Scripts. In Joost-Pieter Katoen and Perdita Stevens (editors), Proceedings of the International Conference on Tools and Algorithms for Construction and Analysis of Systems TACAS'2002 (Grenoble, France), Lecture Notes in Computer Science volume 2280, pages 465-469, Springer, April 2002. Available from http://cadp.inria.fr/publications/Lang-02.html

SEE ALSO

aldebaran(LOCAL), **aut**(LOCAL), **bcg**(LOCAL), **bcg_cmp**(LOCAL), **bcg_graph**(LOCAL), **bcg_io**(LOCAL), **bcg_labels**(LOCAL), **bcg_min**(LOCAL), **bcg_open**(LOCAL), **caesar**(LOCAL), **caesar.adt**(LOCAL), **caesar_hide_1**(LOCAL), **caesar_rename_1**(LOCAL), **evaluator**(LOCAL), **evaluator3**(LOCAL), **evaluator4**(LOCAL), **evaluator5**(LOCAL), **exhibitor**(LOCAL), **exp**(LOCAL), **exp.open**(LOCAL), **generator**(LOCAL), **lnt.open**(LOCAL), **lotos.open**(LOCAL), **mcl**(LOCAL), **mcl3**(LOCAL), **mcl4**(LOCAL), **mcl5**(LOCAL), **projector**(LOCAL), **reductor**(LOCAL), **regexp**(LOCAL), **seq**(LOCAL), **seq.open**(LOCAL), **svl-lang**(LOCAL), **xtl**(LOCAL), **xtl-lang**(LOCAL)

Directives for installation are given in files **$CADP/INSTALLATION_***.

Recent changes and improvements to this software are reported and commented in file **$CADP/HISTORY**.

BUGS

Please report any bug to cadp@inria.fr