Mississippi State University (MSU)
High Performance Computing Collaboratory (HPC^2)
Center for Advanced Vehicular Systems (CAVS)

Developer Installation and Setup Instructions

1. Download the archive file for each SimSys Package that is to be installed.

Go to the SimSys Software Forum and login using your username and password at http://www.simcenter.msstate.edu/software/forum/login.php. Then select the appropriate package link from those at the top of the page. For each package that is available on the forum there are three sub-folders named *archive* (for older versions), *release* (for the current release), and *special* (for special files if available and as needed). Each folder will contain SimSys Package archive files that are self-contained and include all files for a working system. Also, each folder contains descriptive documents such as this one, including README.pdf, LICENSE.pdf, README_install.pdf, and README_packages.pdf.

2. Create a SimSys installation directory.

Create an installation directory for all SimSys Packages, such as "*simsys*". Future upgrades can be installed over existing files or multiple directories can be used for each release/upgrade, e.g. *simsys_release, simsys_sept_rel, simsys_july_rel*, etc. If multiple users will be accessing the files, then a global location would be best. The user installation directory typically is static after the files are installed. There is no need for non-developer individual users to write within the installation directory. A separate developer install should be made for developers that may modify code.

3. Install the SimSys Package files.

The following Linux/MacOSX terminal commands will extract the SimSys Package files.

```
cd simsys (or other installation directory)
tar -zxvpf location_of_package_tar_files/package_archive_file
```

Repeat the command for each package file. Alternatively, you can do the equivalent using the GUI on Linux, MacOSX, or Windows systems. On Windows the files use zip compression. After installing the files, the following directories should exist.

bin/	executables along with dynamic shared libraries for the target system (copied or linked to the system specific directory, such as *Linux-x86-64/bin*)
interface/sm/	GUI definitions (if *SolidMesh* is installed)
*doc/*system/	html-based documentation files for the overall *SimSys* system
doc/aflr2/	html-based documentation files for *AFLR2* (if installed)
doc/aflr2c/	html-based documentation files for *AFLR2C* (if installed)
doc/aflr3/	html-based documentation files for *AFLR3* (if installed)
doc/aflr4/	html-based documentation files for *AFLR4* (if installed)
doc/bloom3 /	html-based documentation files for *BLOOM3* (if installed)
doc/grid_tools/	html-based documentation files for *Grid_Tools* (if installed)
doc/sm/	html-based documentation files for *SolidMesh* (if installed)
doc/ug_io/	html-based documentation files for *UG_IO* input/output files
doc/uvmap/	html-based documentation files for *UVMAP* (if installed)
(+) *sbin/*	developer *sh*-shell script files.
(+) *Linux-x86-64/bin/*	executables and dynamic shared libraries built for Linux-x86-64
(+) *Linux-x86-64/lib/*	libraries of object code built for Linux-x86-64

(+) *MacOSX-x86-64/bin/*	executables and dynamic shared libraries built for *MacOSX* -x86-64
(+) *MacOSX -x86-64/lib/*	libraries of object code built for *MacOSX* -x86-64
(+) *WIN64/bin/*	executables and dynamic shared libraries built for *WIN64*
(+) *WIN64/lib/*	libraries of object code built for a *WIN64*
(+) *src/name/*	source code for "*name*" *specific* libraries, such as *ug, ug2, ug3,* etc.

Similar directories may be added in the future that follow this same directory structure. The above directories, except those preceded by (+), are included in standard package files and contain all files required to run a particular code. Developer packages files are preceded by (+) and named *_LIB.* or *_SRC.* and include all the standard package contents along with object code libraries on supported systems, script files, source code for API related routines, and source code for unrestricted code. Restricted or proprietary package files are typically named for a given site, are encrypted, and contain complete source code for a given program or programs.

4. Configure the SimSys installation.

Script files for developer commands are in the *sbin* developer *sh*-shell script file directory. Developer commands include the following.

simsys_archive	Create package archive of specific SimSys files.
simsys_compile	Compile a SimSys program or library.
simsys_doc	Open documentation in a web browser.
simsys_make_bin	Create a SimSys user bin directory *bin/* (or optional alternative location) with a copy (or optional link) to all SimSys executables and user *sh*-shell script files (and optionally developer *sh*-shell script files).
simsys_www	Create www site package archives for SimSys files (for use at MSU only).

To use the developer commands their path should be included in your system environment PATH. There are two modes available. Choose one of the following operating modes.

a. Create a link to all the files in a location that is already in the environment PATH. Change (*cd*) your working directory to the desired directory that you wish to contain the files and run the following command.

   path.../simsys/sbin/simsys_make_bin -cwd -link -sbin

   Where *path...* is the path to the SimSys installation directory. Note that you may also specify a directory rather than moving to it using the option *–dir name* instead of the option *–cwd*. You may need to rerun this command if you create or install new SimSys executables that were not included in the original installation. Links to the executables, once created, will always point to the actual file so that any subsequent executable updates will be readily available. You do not need to rerun the *simsys_make_bin* command each time an executable is updated. A rerun is needed only if a new executable is added that did not exist when you last created the links with *simsys_make_bin* command.

   If you wish to create a frozen bin file directory that has copies, rather than links, of the current executable and script files then run the *simsys_make_bin* command in the desired directory and use a *-copy* option instead of the *-link* option. Also, if you don't want the developer scripts, other than *simsys_doc*, included then omit the *-sbin* option.

b. Add the locations of the *bin*/ and s*bin*/ directories to your system environment PATH, e.g. *path.../simsys/bin* and *path to SimSys installation directory.../simsys/sbin*. Where *path...* is the path to the SimSys installation directory.

To run a specific program, after configuring your system, use the command line with the name of the specific software followed by options. For example, to test the installation and display a usage summary for *AFLR3,* or any of the other available programs, enter the following command line statement.

```
aflr3 -h
```

If you created a frozen directory or installed other versions of SimSys executables in different locations, then you can access those files using the version directory option. For example, to run another version of AFLR3 that is in directory */other_version...* then use the following command.

```
aflr3 -v /other_version... [options]
```

Enter the following command line statement to test the developer installation and view options for the compile script.

```
simsys_compile -h
```

For example, to update AFLR3 run this command with *aflr3* as the program name.

```
simsys_compile aflr3
```

This will recompile all AFLR3 related files that have changed and rebuild the executable. The script command *simsys_compile* uses standard make files that are generated by the script to update the program.

5. View documentation.

A script is also provided to open the system documentation main page. Enter the following command line statement to use this script.

```
simsys_doc
```

Alternatively, or if the script does not work with your systems then you should point your browser to the system documentation main page file path.../simsys/doc/system/index.html. The system documentation main page has links to all the overall system related information as well as links to documentation for specific packages.

Note that if you chose to move the contents of the *bin/* directory to a directory that is already in the environment PATH (alternative 4. c. above) then the script cannot be used as it will not know the actual location of the *doc/* directory (the script assumes that *bin/* and *doc/* have the same root). With a developer installation, this may not be an issue since *simsys_doc* is also in the *sbin/* directory and it will depend on which location is first in the PATH.

6. Additional comments.

All executables created by the *simsys_compile* script are located in the architecture dependent executable directories *arch/bin*. Libraries are in the architecture dependent library directories *arch/bin*. After code is compiled the developer script *simsys_make_bin* can be used to create copies or links to the executables in the user bin directory *bin/*. For a developer installation, it may be advantageous to use links so that there is no duplication.

The executables (and scripts) located in the *bin/* directory use a naming that is the same as that of the program name. On Linux and MacOSX the executables do not have a suffix if a script is not required. On WINDOWS the executables always have an *.exe* suffix. Names and comments for most of the SimSys programs and scripts are listed in the following table.

Linux/MacOSX name	**WINDOWS name**	**Description**
simsys_doc	*simsys_doc.bat*	Script (*/bin/sh* shell or WINDOWS *batch* file) to open system main documentation page.
aflr2	*aflr2.exe*	*AFLR2* executable (with BL).
aflr2c	*aflr2c.exe*	*AFLR2C* executable (with anisotropic adaptation).
bsurf2	*bsurf2.exe*	*BSURF2* 2D edge grid creation executable.
xplt2	*N/A*	*XPLT2* X-window display for 2D grids executable.
aflr3	*aflr3.exe*	*AFLR3* executable.
aflr3_l	*aflr3_l.exe*	*AFLR3* executable with 64-bit integers.
aflr4	*aflr4.exe*	*AFLR4* executable.
bloom3	*bloom3.exe*	*BLOOM3* executable.
bloom3_l	*bloom3_l.exe*	*BLOOM3* executable with 64-bit integers.
ugc	*ugc.exe*	*UGC* executable.
ugc_l	*ugc_l.exe*	*UGC* executable with 64-bit integers.
uvmap	*uvmap.exe*	*UVMAP* test program executable.
sm	*N/A*	*SolidMesh /bin/sh* shell script required to set the SOLID_MESH_SETUP_DIR environment variable. This variable must be set to the location of the interface/ directory, e.g. path.../simsys/interface/sm. The script can be eliminated if that is made part of your standard environment. The script runs the executable *sm.exe.* Run *SolidMesh* using the script name *sm* or the executable name *sm.exe* if the SOLID_MESH_SETUP_DIR environment variable is set.
sm.exe	*N/A*	SolidMesh executable.
grid_tool_name	*N/A*	*GRID_TOOLS* executable, where *grid_tool_name* is *checkgrid, extract, gridmerge, scalegrid, sgridmerge, surftool,* or *voltol*.
name	*name.exe*	*NAME* other SimSys standard executable.
name_l	*name_l.exe*	*NAME* other SimSys standard executable with 64-bit integers.