SYSTEMA V4.9

Installation Guide

September 2024

Ref: UM.000166345.AIRB
Edition: 5

Use of the software and of the present software manual is submitted to a license agreement to be accepted before the software installation on a computer.

All suggestion or error concerning the software or this software manual can be sent to
systema.business@airbus.com

Table of contents

Introduction

The *SYSTEMA Installation guide* provides instructions on how to install and configure SYSTEMA V4.9.4 and later on Linux and Windows systems. This document assumes that you have a working knowledge of the applicable operating environments.

Please feel free to contact us if you need further information about SYSTEMA. To get how to contact us, please visit https://www.airbus.com/en/space/customer-services/systema#contact

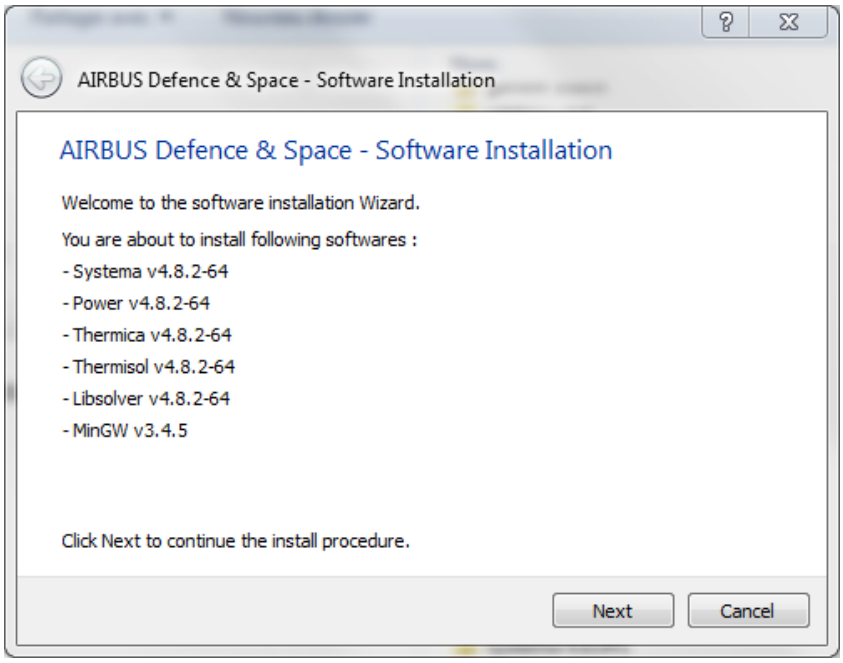
Installing SYSTEMA

Download products

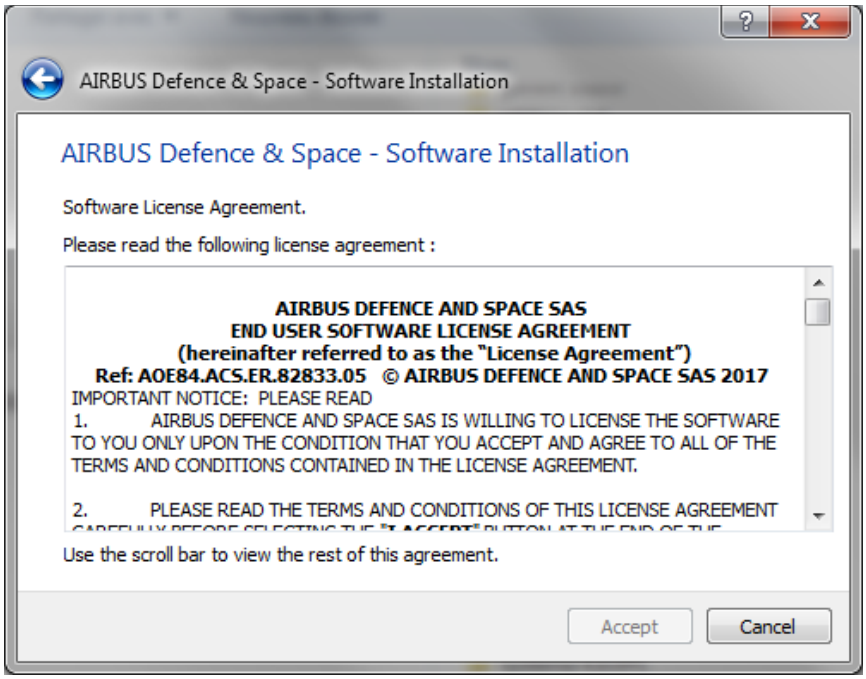
All releases of SYSTEMA are available on demand. Please contact us through https://www.airbus.com/en/space/customer-services/systema#contact.

Installation step-by-step

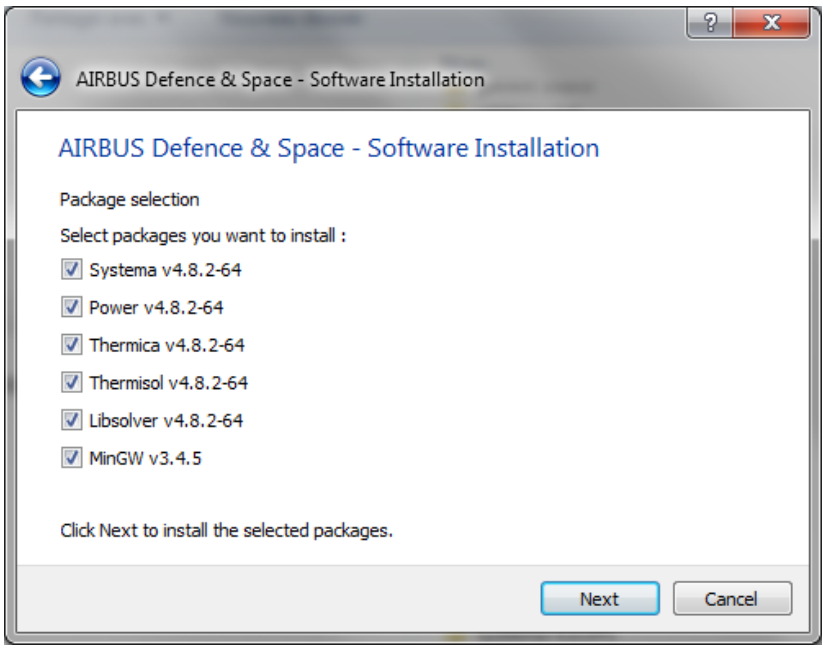
- Start the Systema installer



- Review the Software License Agreement

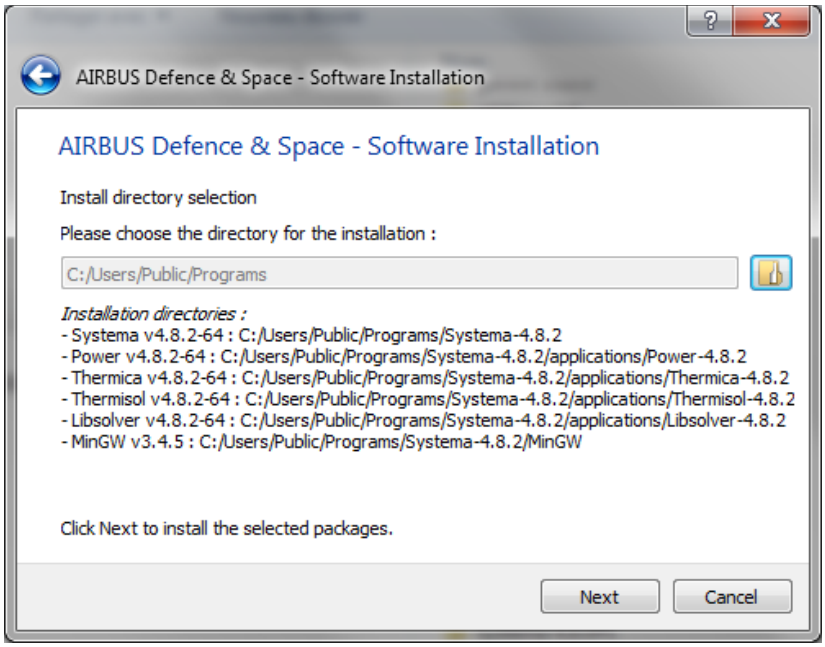


- Specify the packages to install



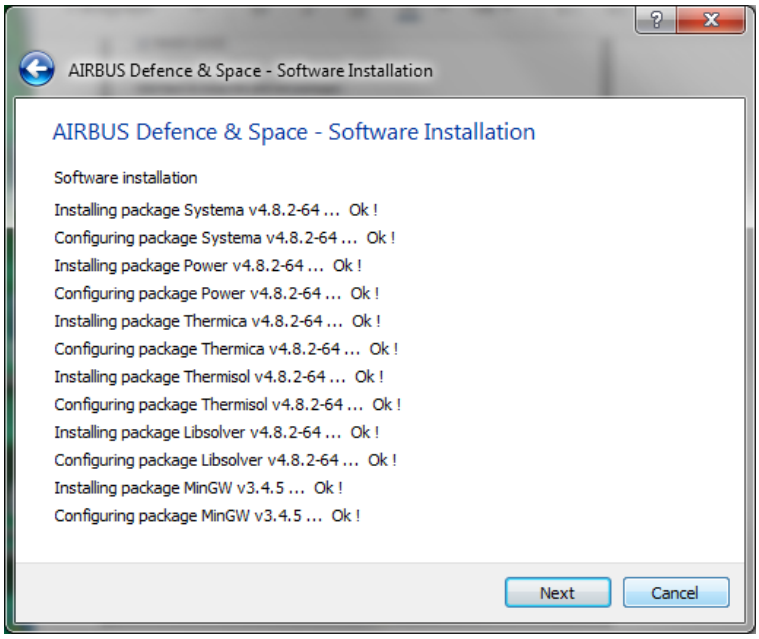
It is highly recommended to keep the default selection.

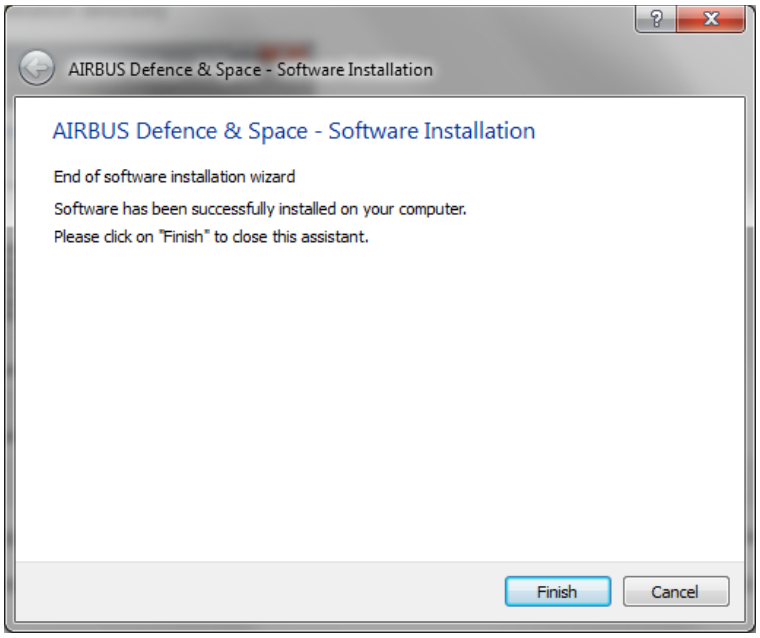
- Select the installation directory



Please avoid the spaces in the Systema folder names, especially in the path where Systema is installed.

- Launch the installation





Any user of Systema shall have sufficient access right to access and execute any files located in the installation folder.

Silent installation

Systema support silent installation that remove the need for user interaction

Systema's silent install standard command line parameters are as follows:

- --silent : Enable the silent installation mode.
- --installDir=<Directory> : Define the parent directory for the Systema installation (Optional).

System requirements

Systema is fully supported on the following OS:

- Windows 7
- Windows 10
- Linux Red Hat 6.6
- Linux Red Hat 8.10

Only 64 bits versions of Systema are available.

For Systema 4.9.4 and above, the following packages are required on Linux:

- libstdc++

To be able to load SYSBAS files on Linux 64bit the following packages have to be installed:

- expat.i686
- fontconfig.i686
- freetype.i686
- glibc.i686
- libICE.i686
- libSM.i686
- libX11.i686
- libXau.i686
- libXcursor.i686
- libXdamage.i686
- libXext.i686
- libXfixes.i686
- libXi.i686
- libXinerama.i686
- libXmu.i686
- libXrandr.i686
- libXrender.i686
- libXt.i686
- libXxf86vm.i686
- libdrm.i686
- libgcc.i686
- libselinux.i686
- libstdc++.i686
- libuuid.i686
- libxcb.i686

- mesa-dri-drivers.i686
- mesa-libGL.i686
- elfutils-libelf.i686
- mesa-dri-filesystem.i686
- mesa-private-llvm.i686
- mesa-libGLU.i686

SYSTEMA Licensing

Introduction

SYSTEMA requires a valid SYSCOD V5 license to be fully operational. Two licensing mode are support by SYSTEMA :

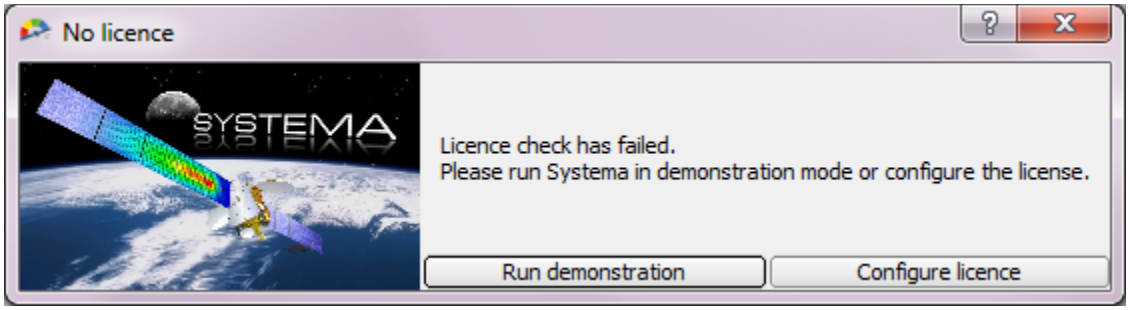
- The “local license” system is used to provide authorization to execute applications without using any floating licenses (authentication through a license server). The benefit of this system is that a license server is not required. All information to make applications work are defined in a license file. This file’s name must be suffixed by .moslf to be recognized as a license file by the applications using SYSCOD V5.
- The “license server” system is used to provide the authorization to execute applications on any computers to the network. The benefit of this system is that only one license file is required, containing all needed information to make the application to work. This file’s name must be suffixed by .moslf to be recognized as a license file by the applications using SYSCOD V5.

SYSCOD V5 is available on Windows and Linux (see § Download Systema). For more detail about this license manager, please refer to "*SYSCOD V5 - User manual*" (available on https://www.airbus.com/sites/g/files/jlcbta136/files/2023-02/SyscodV5_InstallationManual.pdf).

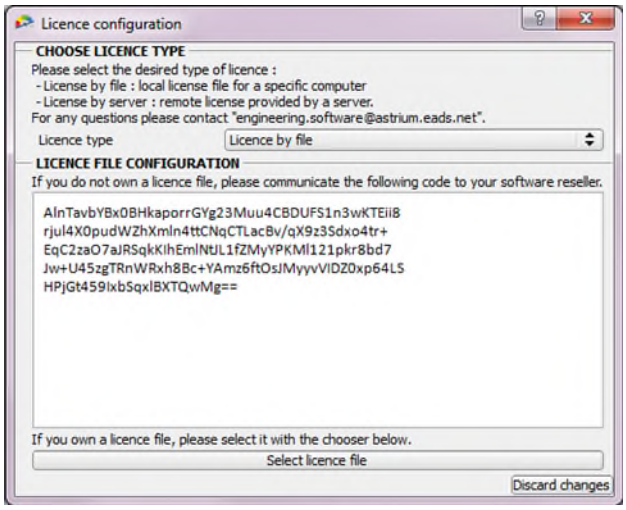
Configuration

Once downloaded and installed, to get a local license please:

- Launched SYSTEMA
- Choose **Configure licence** from the popup window.



- A new window will open displaying a crypted key. Please copy paste this key and send it to our technical support team (see Introduction) so that we will generate a license file in turn.



For server licenses, please contact the technical support team.

Troubleshooting

Issue	Solution
When using Systema through a ssh connexion, the GUI is often refreshed.	Set `QT_GRAPHICSSYSTEM = native` 3 graphics backend are supported by Qt4(.5+) : *native*, *raster,opengl*. *native* must be used when using a SSH X-foward mode whereas the other ones must be used for a local rendering
All 3D views of Systema are empty	Please check that the OpenGL rendering is allowed by your system

3D problem through SSH Fowarding :
Running OpenGL bases applications on a remote computer through a ssh tunnel with X11 fowarding might not be successful and result a X error.
The OpenGL base rendering application may also complain that it is unable to create a GL context.
	Verify that Direct rendering is enabled. You can check this by running
	`glxinfo | grep rendering`
	The Nvidia libGL (at least by defaut) does not allow indirect GLX calls. Preventing the application that uses OpenGL to be run on the remote computer through the secure shell with X11 fowarding and authentification
	Make sure that both the local and the remote machines are using mesa's libGL.
	One way to figure out which libGL implementation the system is currently using is by running
	`ldconfig -p | grep libGL.so`
	Installing mesa library over Nvidia driver could mess up the libGL used by the host.
	A quick fix is copying the libGL.so of /usr/lib/x86_64-linux-gnu/mesa from hother host. Then, once ssh to the host, set LD_LIBRARY_PATH as follow :
	`export LD_LIBRARY_PATH=/usr/lib/x86_64-linux-gnu/mesa:${LD_LIBRARY_PATH}`
	The the current session will use mesa's libGL.so instead of the Nvidia one

Some applicative modules report a crash	Check that there is no restriction on all Systema folders for reading and executing.
"Unable to generate mission file" raised when running a processing linked to a schematic file	Check that no errors about unknown components were raised when opening the schematic file (see log area). If it is the case, rebuild the schematic in a new file.

When displaying a 3D on Linux, the following errors are reported :
`libGL: OpenDriver: trying /usr/lib64/dri/tls/swrast_dri.so`
`libGL: OpenDriver: trying /usr/lib64/dri/swrast_dri.so`
	This issue can be raised on unsupported versions of Linux. A workaround is to copy the libstdc++.so.6 in the Systema lib directory (or rename the libstdc++.so.6 which is in the Systema lib directory so that it is the libstdc++.so .6 of the system that is used).

Issue	Solution
When running applicative modules, some times defined in the mission file cannot be found in HDF5 output files.	When loading mission data, applicative modules optimize the list of computation points by deleting points that are too close to each other. This check is based on the following threshold : $$\frac{\text{last\_mission\_time} - \text{first\_mission\_time}}{8 * \text{nb\_mission\_times}}$$
Launching application return a 'Fail to execute the process' status without any other information	Verify that the ports used by the applications are not busy. The ports can be set by the environment variable : • MOS_TCPMSG_PORT_START : Initial port number used by application (default is 48557) • MOS_TCPMSG_PORT_RANGE : How many ports from MOS_TCPMSG_PORT_START an application will try to use
The application crash with a Java error reporting that there is insufficient memory for the Java Runtime Environment to continue.	Running multiple systema applications in the same time can an out fo memory error. To reduce memory consumption by application, set the environement variable _JAVA_OPTIONS as follow : `_JAVA_OPTIONS =-Xms1024m -Xmx1024m`

SYSTEMA environment

Files organization

Once the SYSTEMA application has been installed in your system, the installation directory contains:

- the main SYSTEMA binary file (with name depending on the operating system):

File name	Operating system
SystemaWIN.exe	Windows
SystemaLNX	Linux

- three subdirectories containing additional binaries and data files:

Directory	Content
applications	For each application interfacing with SYSTEMA, it contains a specific subdirectory with all the necessary binary and data files
bin	All the necessary binary files for SYSTEMA use
data	All the internal data files, such as image files
licenses	All the licenses files for SYSTEMA use, if necessary
doc	Contains SYSTEMA documentation files
python	All the python script files for SYSTEMA use

The SYSTEMA working directory

The SYSTEMA application needs a working directory which will be used by default when creating files and searching files.
The SYSTEMA working directory can be defined via the environment variable "**SYSTEMA_WORKING_DIR**". If not, the working directory is the user home directory ("my documents" for PC system, "Home directory" for UNIX system).

Environment variables

SYSTEMA uses two environment variables which have to be set before running the application:

- **"SYSTEMA_HOME"**, defines the directory containing the SYSTEMA executable file and the **applications**, **bin** and **data** directory.
- **"SYSTEMA_WORKING_DIR"**, defines the directory that will be used by default to store created files and generated results; this variable is optional.

SYSTEMA also use a set of configurable environment variables:

- To manage SYSTEMA :

Variable	Definition
SYSTEMA_DISABLE_3D_OPTIM = 1	Disable 3D optimizations (useful in case of graphics hardware incompatibility for 4.8.2+ versions). *Required for OpenGL <= 2.0*

	3D optimization can also be managed with the settings
SYSTEMA_NOAUTO3D = 1	Deactive the automatic opening of 3D views when opening a file
SYSTEMA_SATRATIO = n	Customized satellite size in the 3D view and access to real size of the celestial body (by default 1000)
SYSTEMA_FONTSIZE = n	Size of the display font This option can also be managed with the settings
SYSTEMA_STEPTAS = 1	Active the import and export of Step-Tas (version < 4.3.3c)
SYSTEMA_BDF_IMPORT = 1	Active the bdf/blk import (version < 4.3.3c)
SYSTEMA_BDF_FIXED_DOUBLE = 1	Export bdf/blk: 'fixe double' format instead of 'libre double'
SYSTEMA_BDF_THERMALID_AS_PID = 1	Export bdf/blk: thermical id exported as property id
SYSTEMA_BDF_GRIDID_START = 1	Export bdf/blk: value of the first grid id
SYSTEMA_BDF_ELEMID_START = 1	Export bdf/blk: value of the first element id
SYSTEMA_ANTI_ALIASING = 1	Enable anti-aliasing
SYSTEMA_SOFTWARE_RENDER = 1	Disable 3D acceleration by graphics card
SYSTEMA_2DGT_RESOLUTION = n	Maximum resolution of arcs in the ground track view (by default 100000)
SYSTEMA_ATT_NOSIGN = 1	Deactivate the icon triangles for performance
SYSTEMA_TEXTURE_HR = 1	Activate the high resolution textures
SYSTEMA_NO3DEFFECT = 1	Disable the planet indicators and the sun effect
SYSTEMA_NOAUTOTIMELINE = 1	Disable the default opening of the timeline in the trajectory and the mission
SYSTEMA_EDITORDOCKED = 1	Automatic lock of edition window
SYSTEMA_STL = 1	Activate the STL export feature
SYSTEMA_STP_CHORD_TOLERANCE = c	Overloads the value of the chord tolerance used for step files tesselation (by default c = 1 mm) This option can also be managed with the settings
SYSTEMA_STP_ANGLE_TOLERANCE = a	Overloads the value of the angle tolerance used for step files tesselation (by default a = 60 deg) This option can also be managed with the settings
SYSTEMA_PROFILES_DIR = *path*	Define the *path* where the profiles are stored
SYSTEMA_NTHREAD = n	Number of threads for the multithreading computation (only applicable for Thermica and PlumImp)
SYSTEMA_COMPONENTS_PATH = *path*	Define the *path* where the components for the schematic are stored
SYSTEMA_DEFAULT_PYTHON_CONSOLE = 1	Use the default Python console instead of the Jupyter console
SYSTEMA_ENABLE_HIGH_RES = 1	Enable the maximum resolution to 8k screen

- To manage Scheduler:

Variable	Definition
MOS_TCPMSG_PORT_START = s	Start of the range of ports used by the network (by default 48557)
MOS_TCPMSG_PORT_RANGE = r	Area of the range of ports used by the network (by default 17 for version < 4.4.2, then 3 from version 4.4.2WIN)
SYSTEMA_CONNECTING_ATTEMPTS = c	Number of connecting attemps on each port of the list (by default 3)
SYSTEMA_TIMEOUT_CONNECTION = t	Timeout limit for each attempt on each port in ms (by default 3000 from version 4.4.1)