Stood

Start Up Guide

Revision F

Ellidiss Technologies
http://www.ellidiss.com

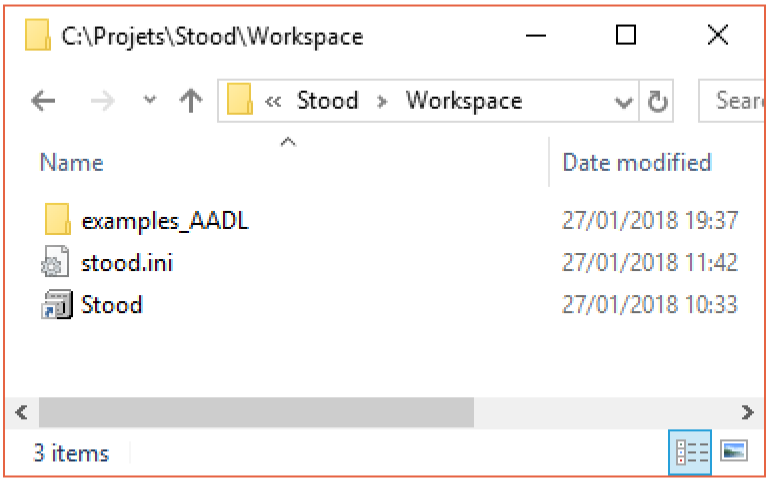
After installing the software and its license, **Stood** is ready to use, and all its features for **AADL** or **HOOD** architectural and detailed design activities are enabled. This document provides some basic information on using the tool so that you can get started. To get help for the product installation process, please refer to the *Stood Install Manual*.

This quick start guide assumes that you are using a standard standalone installation on a **Windows PC**. Please note that there may be a few differences if you are using a **Unix** version, or a network installation.

1 Launch Stood

The standard installation creates two shortcuts for **Stood** in the **Windows** startup menu (*Stood for HOOD* and *Stood for AADL*), and one on the desktop (*Stood for AADL*). The default working directory (workspace) that is defined by the installation program is `{Current_User_Directory}\Documents\Stood`. This default workspace contains a set of predefined examples that are described at chapter 2.

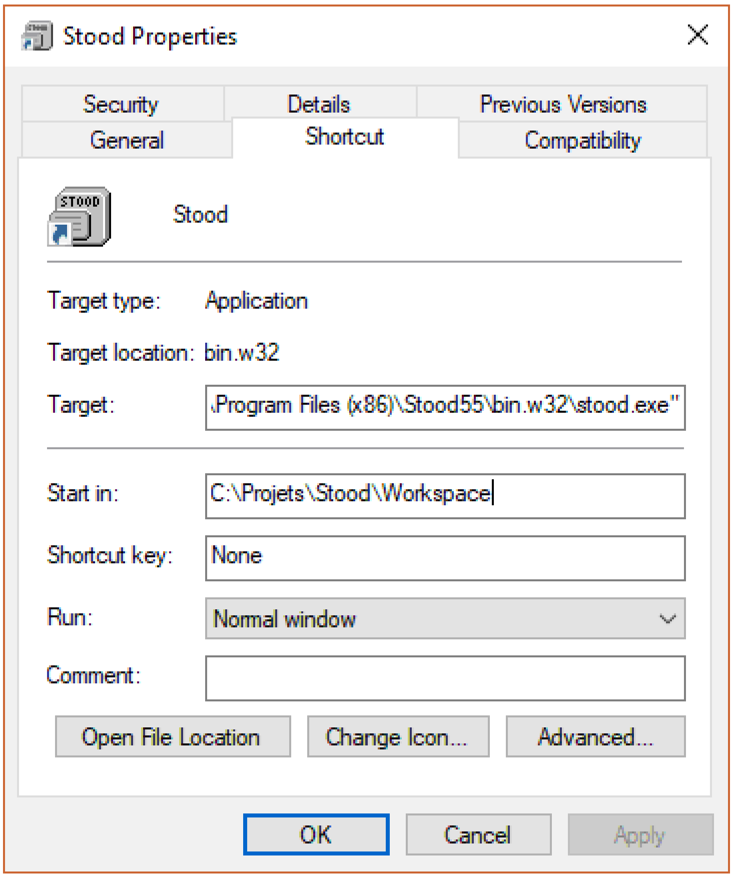
You can create other working directories anywhere else on your system. To do so, create a new directory, ensure that you have all the necessary access rights within this directory, and add the following items into it:
- A shortcut of the **Stood** executable (*Stood for HOOD* or *Stood for AADL*).
- A local initialization file (*stood.ini*).
- Optionally, a copy of the predefined examples.


C:\Projets\Stood\Workspace

Name	Date modified
examples_AADL	27/01/2018 19:37
stood.ini	27/01/2018 11:42
Stood	27/01/2018 10:33

3 items

Edit the shortcut properties and set the *Start in* field to the working directory you just created. This implies that each time **Stood** is executed using this shortcut, any newly created **Projects** and **Designs** will be stored in your local working directory.



Target type: Application

Target location: bin.w32

Target: .Program Files (x86)\Stood55\bin.w32\stood.exe"

Start in: C:\Projets\Stood\Workspace

Shortcut key: None

Run: Normal window

Comment:

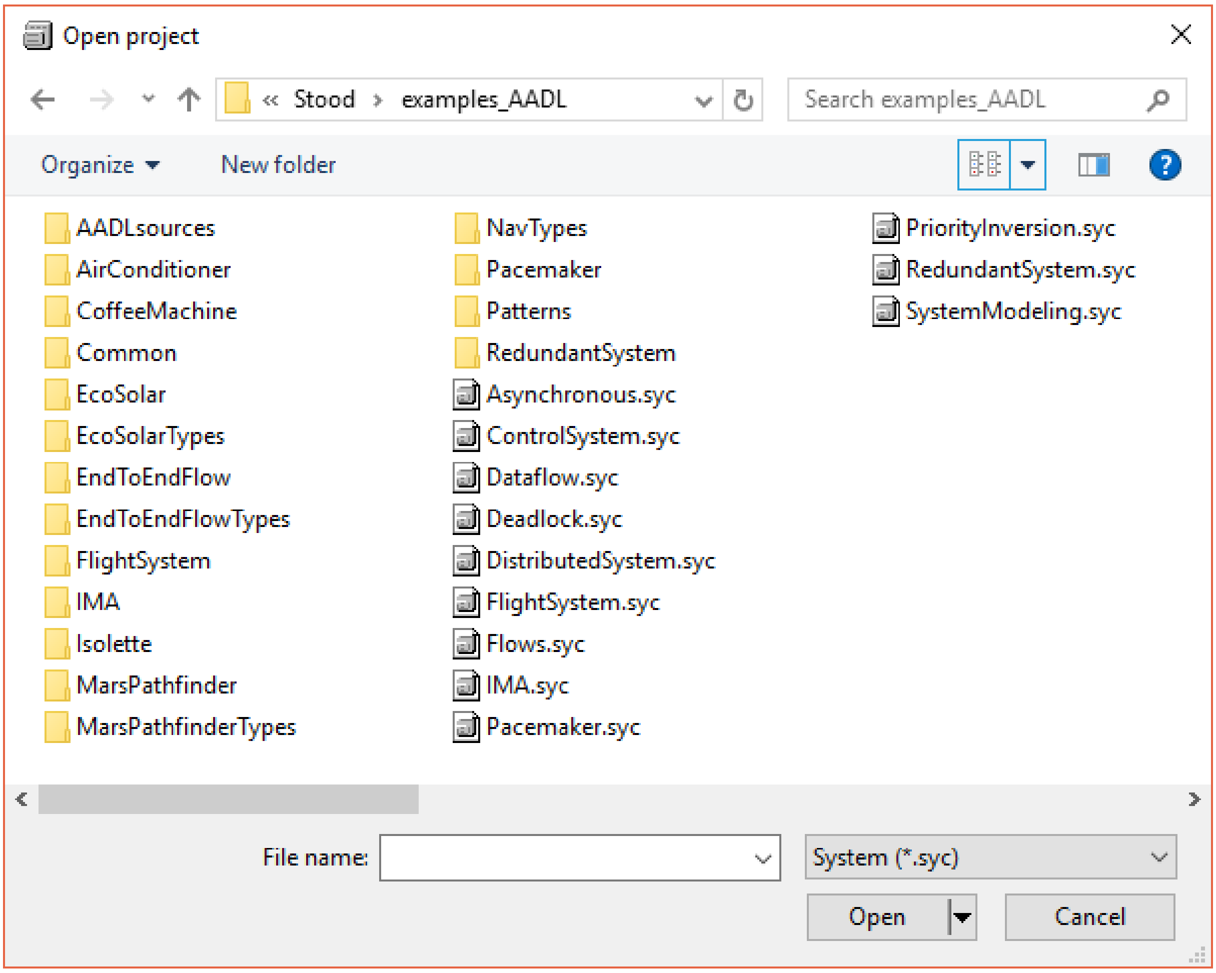
The local initialization file (`stood.ini`) may be created and edited with any text editor. This file should only contain the properties that differ from the standard configuration. These properties will overload the default ones if **Stood** is executed from this working directory. For instance, this local `.ini` file may typically contain the following customized sections:

```
[General]
Welcome=My Stood workspace

[Environment]
PROJECT=My_Project
COMPANY=My_Company
OSATE_PATH=/cygdrive/C/Osate/osate.exe
AI_PATH=/cygdrive/C/AI-1.6/bin/AADLInspector.exe

[ConfigurationManagement]
Versions=Draft,SRR,PDR,CDR,AR

[ETLicense]
ServerName=192.168.1.201
ServerPort=29000
Id=Ellidiss
```

2 View the examples

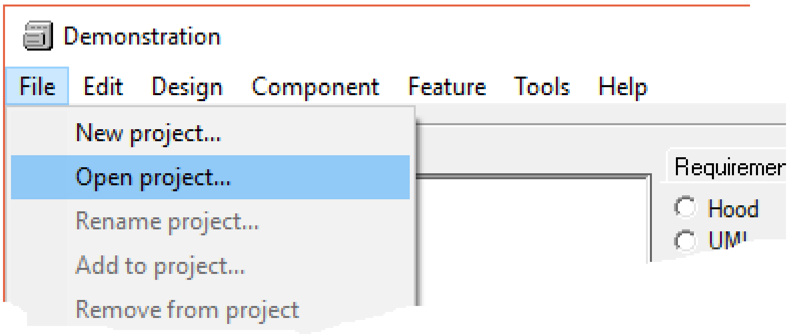
To view the predefined examples that are stored in the default workspace, launch **Stood** from the Start Up menu or the shortcut that was created on the Desktop during installation. If you want to view the predefined examples from within another workspace, please copy the example directories into this location first and launch **Stood** with the shortcut you created in this workspace.

Each example is identified by a `.syc` file that defines the list of **Designs** that collaborate to a given **Project**. A same **Design** may collaborate to several **Projects**.



AADLsources
AirConditioner
CoffeeMachine
Common
EcoSolar
EcoSolarTypes
EndToEndFlow
EndToEndFlowTypes
FlightSystem
IMA
Isolette
MarsPathfinder
MarsPathfinderTypes
NavTypes
Pacemaker
Patterns
RedundantSystem
Asynchronous.syc
ControlSystem.syc
Dataflow.syc
Deadlock.syc
DistributedSystem.syc
FlightSystem.syc
Flows.syc
IMA.syc
Pacemaker.syc
PriorityInversion.syc
RedundantSystem.syc
SystemModeling.syc

To load one of the examples, use the *Open project…* option in the *File* menu:



This opens a standard file selector. Navigate to the `examples` directory and select a `.syc` file. This will load the corresponding example. The directories `examples` and `examples_libs` contain **HOOD-Ada**, **HOOD-C/C++** designs and libraries, whereas the directory `examples_AADL` contains ten illustrative examples of graphical **AADL** models. For each of them, the complete set of textual **AADL** code can be generated and analysed with the **AADL Inspector** companion tool.

Example 1. "Dataflow.syc": dataflow inter-threads communication
in AADL Inspector, check the effect of sampled, immediate or delayed connections

Example 2. "Deadlock.syc": possible deadlock situation
in AADL Inspector, check the effect of using Priority Ceiling Protocol or not

Example 3. "PriorityInversion.syc": possible priority inversion situation
in AADL Inspector, check the effect of using Priority Ceiling Protocol or not

Example 4. "Asynchronous.syc": user interaction with a simulated coffee machine
in AADL Inspector, use the process I/O control panel

Example 5. "DistributedSystem.syc": CAN bus based distributed architecture of a solar vehicule
in AADL Inspector, observe the impact of the bus communication on threads scheduling

Example 6. "ControlSystem.syc": an automated heat regulator
in AADL Inspector, specify a desired temperature and observe the behavior of the regulator

Example 7. "RedundantSystem.syc": three sensor failure detection system
in AADL Inspector, observe how the input selection logic works

Example 8. "SystemModeling.syc": architectural design of an isolette with requirements traceability

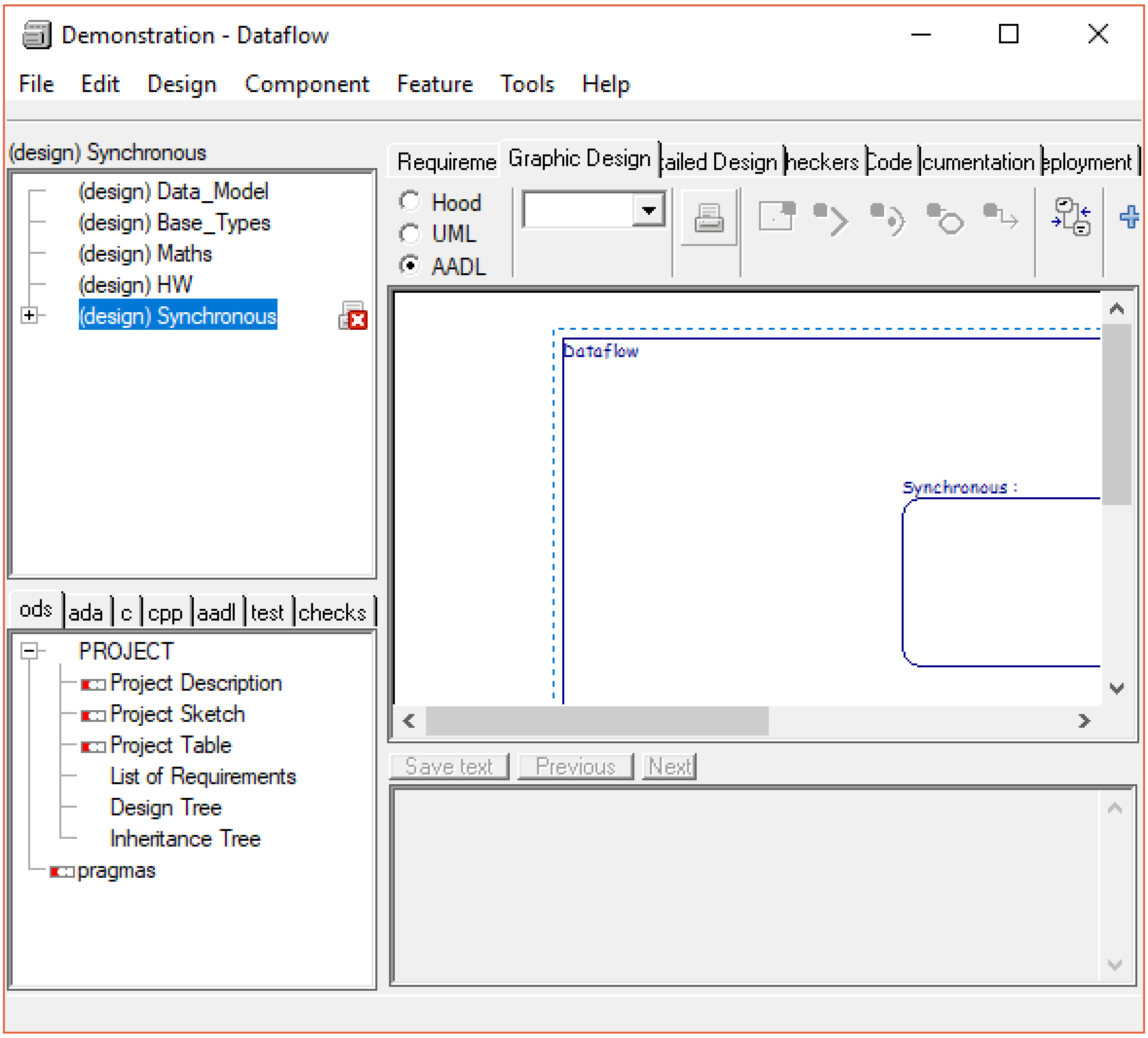
Example 9. "FlightSystem.syc": simplified aircraft flight system.

Example 10. "IMA.syc": another flight control system deployed with an Integrated Modular Architecture.

Example 11. "Pacemaker.syc": pacemaker control software.
in AADL Inspector, test scenarios can be used with the simulator.

Example 12. "Flows.syc": automatic generation of AADL flows and end to end flows.

After a `.syc` file has been loaded, you must select one of the listed **Designs** to be able to edit it. When selected, a **Design** is loaded in read-only mode by default. It is thus possible to navigate within the **Components** hierarchy without altering it. To be able to perform changes, the **Design** must be firstly locked (see details in next chapter).



Demonstration - Dataflow

3 Create a new Design

A **Project** is composed of several sharable **Designs** (your current application, reusable components, libraries, other sub-systems, ...). When opening **Stood** for the first time, only the predefined **Project** examples are available. To create a new **Project**, select the *New project...* option from the *File* menu and give it a name in the displayed dialog box:

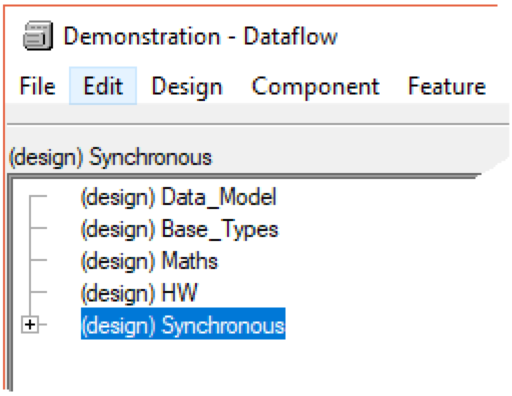
The created **Project**, however does not contain any data yet. To reference already existing **Designs**, select the *Add to project ...* option from the *File* menu, and select the chosen `Stood.sto` file from the file selector.

To create a new **Design**, select the *New design ...* option from the *Design* menu. Several options are available:

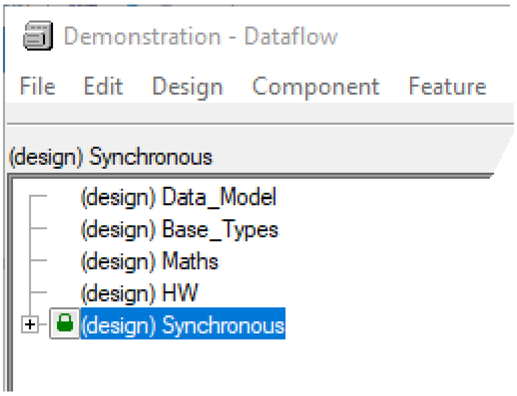
- **HOOD** designs: select *design*, *generic* or *virtual node*
- **AADL** designs: select *package*, *system*, *process* or *processor*

The corresponding entry will appear in the **Designs** list of the **Project**.

A single click on the name in the list, loads the **Design** in read-only mode. To load it in read-write mode, double-click the name or select the *Lock design* option from the *Design* menu. When loaded in read-write mode, a green padlock is shown near the **Design** name. This means that nobody else can open it in read-write mode.



loaded "*read only*"



loaded "*read write*"

The software design activities may now start. The following features are supported by **Stood 5.5**:

- import software requirements (text files or Reqtify 2018)
- reverse engineer legacy **AADL** models, or **Ada** or **C** source code
- perform architectural design activities graphically
- achieve complete detailed design and coding
- apply internal or remote model verification tools
- use code generators to producer **AADL**, **Ada**, **C** or **C++** source code
- generate design documentation in **HTML**, **PDF**, **RTF**, **ODT**,... .

When the **AADL** code generator is used, the embedded **AADL Inspector** companion tool may be automatically launched. **AADL Inspector** must be installed separately and provides a set of **AADL** processing tools such as **Cheddar** for schedulability analysis and **Marzhin** for dynamic run-time simulation. To associate the two tools, **AADL Inspector** install path must be specified in the `AI_PATH` variable of the customized `stood.ini` file (see chapter 1).

Please consult the provided documentation for further details:

- STOOD-Install Manual (Installation Guide)
- STOOD-AADL Manual (Stood for AADL User Manual)
- STOOD-AADL Tutorial (Building an AADL project step by step with Stood)
- AI User Manual (AADL Inspector User Manual)
- STOOD-Admin Manual (Advanced User Customization Guide)
- STOOD-Ada Coding Manual (Ada code generator options)
- STOOD-C Coding Manual (Guidelines for building C software with Stood)
- HOOD-Guidelines (Overview of the HOOD software design method)

In addition, other resources such as on line tutorials can be found on the support web site: http://www.ellidiss.fr

Ellidiss
Technologies

www.ellidiss.com

Sales office:
TNI Europe Limited
Triad House
Mountbatten Court
Worall Street
Congleton
Cheshire
CW12 1AG
UK
info@ellidiss.com
+44 1260 291 449

Technical support :
Ellidiss Technologies
24 quai de la douane
29200 Brest
Brittany
France

aadl@ellidiss.fr
+33 298 451 870