EPRI | ELECTRIC POWER RESEARCH INSTITUTE

OpenDSS Training Workshop

Interfacing OpenDSS and introduction to OpenDSS-G

Davis Montenegro
EPRI Knoxville, TN

August 25, 2020

Instructor

- **Davis Montenegro, *Senior Member, IEEE***

Davis Montenegro-Martinez serves as technical leader at the Electric Power Research Institute (EPRI) in the areas of power system modeling, analysis and high-performance computing. He received his degree in electronics engineering from Universidad Santo Tomás, Bogotá, Colombia (2004); he is M.Sc. in electrical engineering from Universidad de los Andes, Bogotá , Colombia (2012). He received his Ph.D. in electrical engineering from Universidad de los Andes (2015), and a Ph.D. in electrical engineering from the University Grenoble-Alpes, France (2015).

Before joining EPRI, Davis served for 10 years as a lecturer for Universidad Santo Tomas in Colombia, during this time he was also technology consultant in the areas of industrial automation, software and electronic hardware design focused in the electric power industry, specifically in monitoring and control for meter calibration laboratories. His expertise in parallel computing techniques is being used at EPRI for incorporating multi-core processing to power system analysis methods such as QSTS, reducing the computational time required to perform these analysis using standard computing architectures

The evolution of OpenDSS into a parallel computing machine

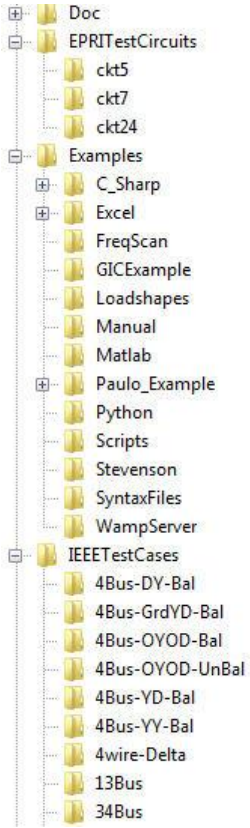
After being released in 2008 as open source software OpenDSS has become widely used around the world. One of the features that makes OpenDSS popular is that the package offers interfaces for co-simulation.

Interfacing with OpenDSS

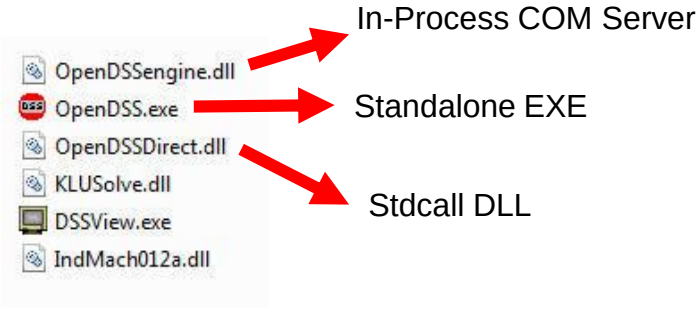
User Interfaces

- A **stand-alone executable** program that provides a text-based interface (multiple windows)
- An **in-process COM server** (for Windows) that supports driving the simulator from user-written programs.
- A **direct DLL** interface that mimics the COM interface
  - For non-Windows platforms, such as HPCs
  - For programming languages that do not support COM or are not efficient at supporting COM

OpenDSS Files Installed

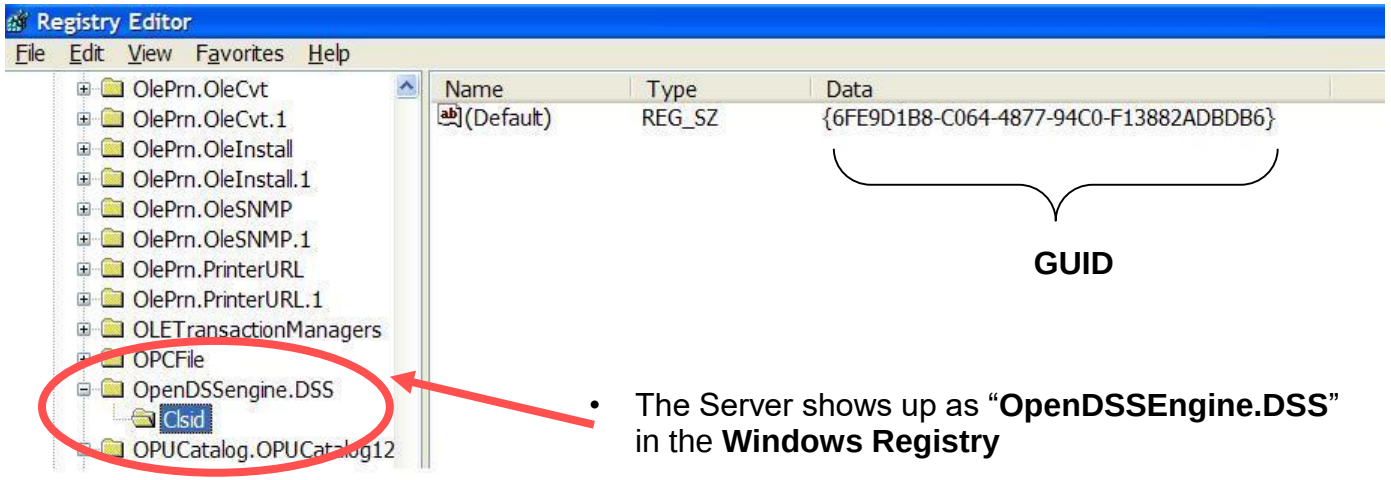


Main Program Files



Registering the COM server

Windows Registry Entry

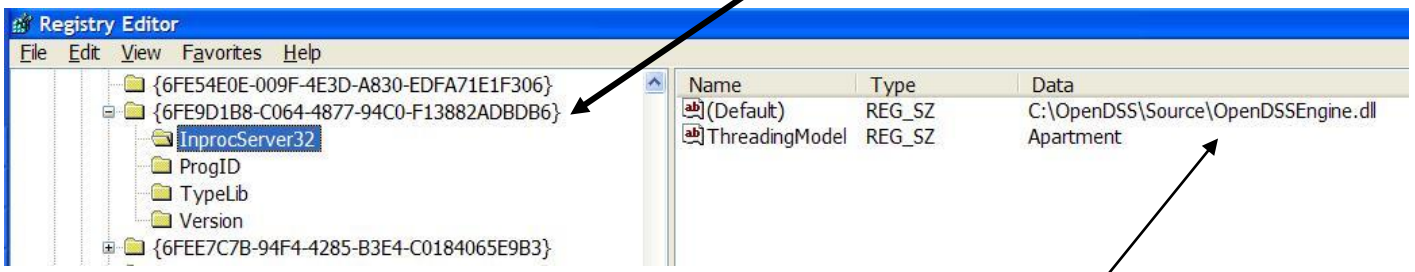


- The Server shows up as "**OpenDSSEngine.DSS**" in the **Windows Registry**

The OpenDSS is now available to any program on the computer

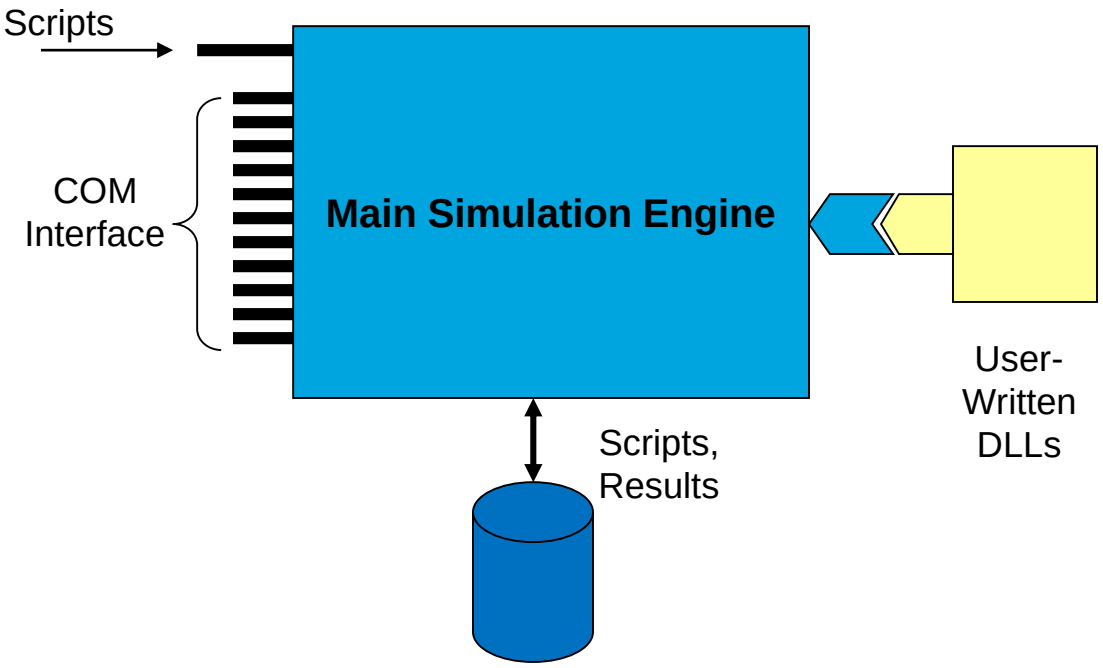
The GUID References the DLL File

If you look up the GUID in RegEdit

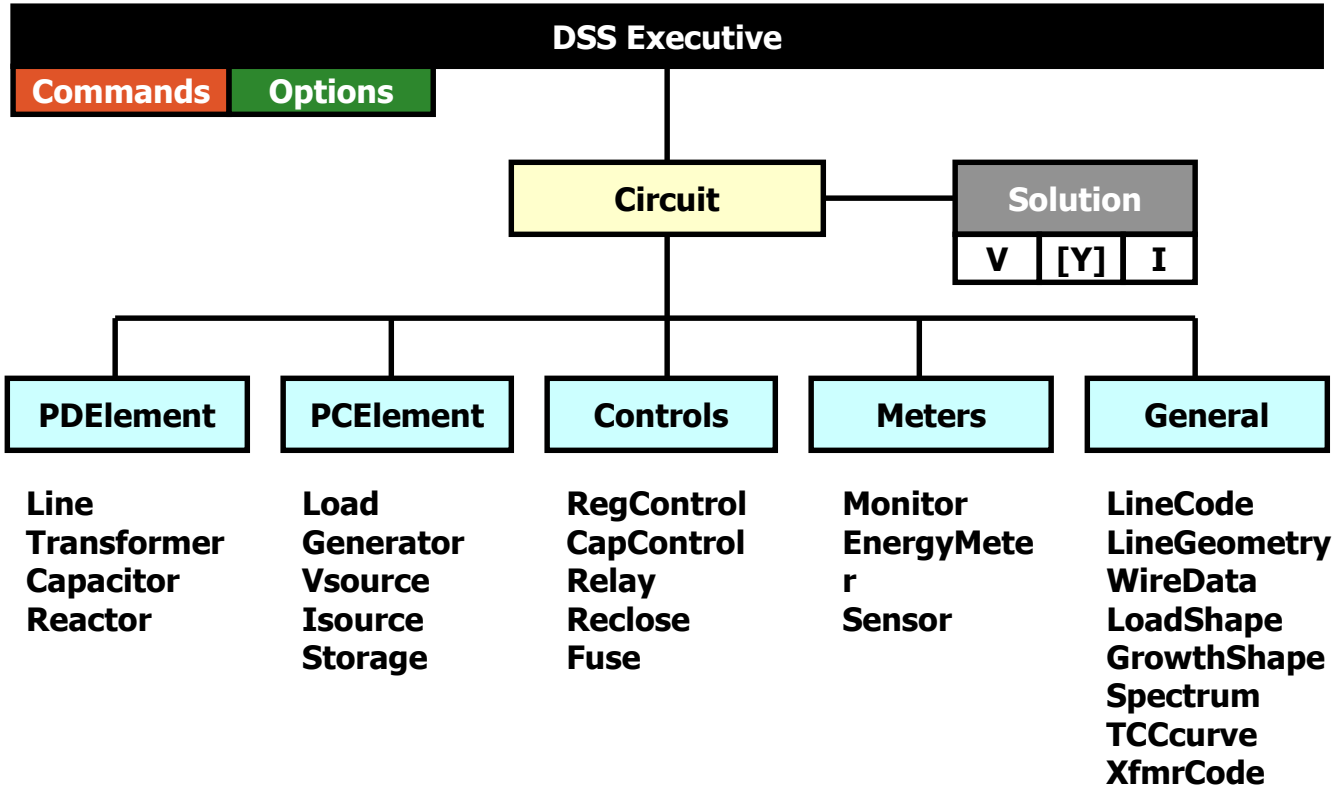


Points to OpenDSSEngine.DLL
(In-process server, Apartment Threading model)

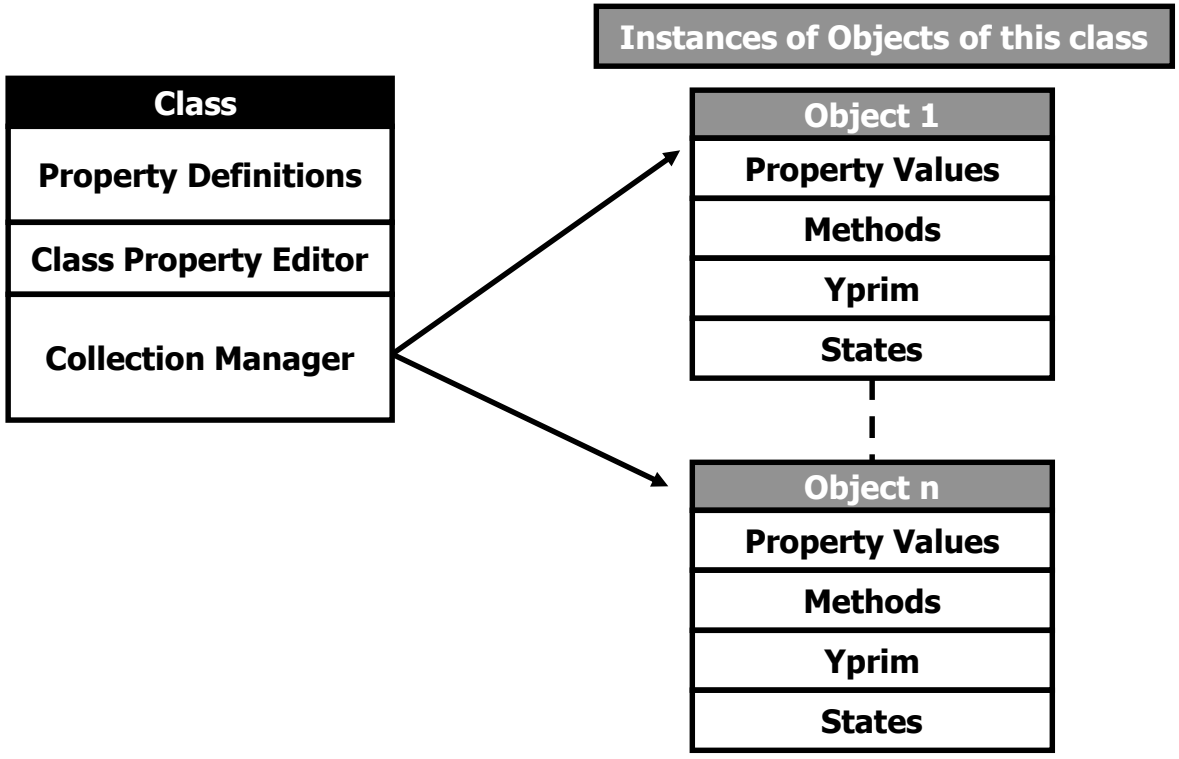
DSS Structure



DSS Object Structure



DSS Class Structure



How can I query about the available interfaces, properties and methods

User Interfaces

- Using a program language that depicts the interface for you (e.g. MS excel, VBA, etc.).

- Reading the documentation: https://sourceforge.net/p/electricdss/code/HEAD/tree/trunk/Version8/Distrib/Doc/

- Using the query tools available in your programming language:

  MATLAB : get, properties…

  Python: getattr, getAllAttributeNames…

- If working with DirectDLL, then, you'll have to read the documentation and probably use the header file provided.

Examples on using COM

Parallel processing in OpenDSS using COM

The examples for parallel processing can be downloaded from:

https://sourceforge.net/p/electricdss/code/HEAD/tree/trunk/Version8/Distrib/Examples/Parallel_Processing/

These examples involves MATLAB and python.

Other examples for LabVIEW through the VI Package Manager

Parallel processing in OpenDSS using COM

Example # 1 MATLAB

```
clc;
[DSSStartOK, DSSObj, DSSText] = DSSStartup;
DSSCircuit      =   DSSObj.ActiveCircuit;
DSSText.Command =   'ClearAll';             % Clears all instances of OpenDSS-PM
DSSText.Command =   'Set Parallel=Yes';             % Clears all instances of OpenDSS-PM

DSSParallel     =   DSSCircuit.Parallel;    % Habdler for Parallel processing functions
CPUs            =   DSSParallel.NumCPUs;    % Gets how many CPUs this PC has
% By default one actor is created by default, if you want more than one
% parallel instance you will have to create them. Try to leave at least
% One CPU available to handle the rest of windows, otherwise will block
% Everything
for i=1:CPUs-1,
    if i ~= 1,
        DSSParallel.CreateActor; % Creates additonal actors
    end;
    DSSText.Command =   'compile (C:\Program Files\OpenDSS\EPRITestCircuits\ckt5\Master_ckt5.DSS)';
    DSSCircuit.Solution.Solve;
    DSSParallel.Wait;   % for the first solve, it is needed to wait before creating other actor
    DSSText.Command =   'set mode=Time stepsize=1h number=16000';
end;
% Now the actors are solved

DSSCircuit.Solution.SolveAll;
```

Parallel processing in OpenDSS using COM

Example # 1 MATLAB

```
25
26  pause(0.1);
27  BoolStatus      =   0;
28  while BoolStatus == 0,
29      ActorStatus     =   DSSParallel.ActorStatus;
30      BoolStatus      =   all(ActorStatus & 1); %Checks if everybody has ended
31      ActorProgress   =   DSSParallel.ActorProgress;
32      clc;
33      for i=1:CPUs-1,
34          fprintf('Actor %i Progress(%%) @ CPU %i : %i\n',i,i-1,ActorProgress(i));
35      end;
36      pause(0.5);  %  A little wait to not saturate the Processor
37  end;
38  disp('Simulation finished by all the actors');
```

Parallel processing in OpenDSS using COM

Example # 2 MATLAB

```
clc;
[DSSStartOK, DSSObj, DSSText] = DSSStartup;
DSSCircuit      =   DSSObj.ActiveCircuit;
DSSText.Command =   'ClearAll';             % Clears all instances of OpenDSS-PM
DSSText.Command =   'Set Parallel=No';      % Deactivates parallel processing

DSSParallel     =   DSSCircuit.Parallel;    % Habdler for Parallel processing functions
CPUs            =   DSSParallel.NumCPUs;    % Gets how many CPUs this PC has
% By default one actor is created by default, if you want more than one
% parallel instance you will have to create them. Try to leave at least
% One CPU available to handle the rest of windows, otherwise will block
% Everything
% Prepares everything for a yearly simulation using temporal parallelization
YDelta  =   8760/(CPUs-1);
disp('Compiling and creating Actors');
for i=1:CPUs-1,
    if i ~= 1,
        DSSParallel.CreateActor; % Creates additonal actors
    end;
    DSSText.Command =   'compile (C:\Program Files\OpenDSS\EPRITestCircuits\ckt7\Master_ckt7.DSS)';
    DSSCircuit.Solution.Solve;
    if i == (CPUs-1),
        YDelta = 8760 - (CPUs-2)*YDelta;
    end;
    DSSText.Command =   ['set mode=Yearly number=',int2str(YDelta),' hour=',int2str((i-1)*YDelta)];
end;
% Now the actors are solved
DSSText.Command =   'Set Parallel=Yes';             % Activates parallel processing
DSSCircuit.Solution.SolveAll;
```

Parallel processing in OpenDSS using COM

Example # 2 MATLAB

```
30
31  pause(0.1);
32  BoolStatus      =   0;
33  while BoolStatus == 0,
34      ActorStatus     =   DSSParallel.ActorStatus;
35      BoolStatus      =   all(ActorStatus & 1); %Checks if everybody has ended
36      clc;
37      % Prints the current time on each simulation
38      for i = 1:(CPUs-1),
39          DSSParallel.ActiveActor = i;
40          CHour   =   DSSCircuit.Solution.dblhour;
41          fprintf('Actor %i Time(hours) : %f\n',i,CHour);
42      end;
43      pause(0.5);  %  A little wait to not saturate the Processor
44  end;
45  disp('Simulation finished by all the actors');
46
```

Parallel processing in OpenDSS using COM

Example # 3 MATLAB

```
clc;
[DSSStartOK, DSSObj, DSSText] = DSSStartup;
DSSCircuit      =   DSSObj.ActiveCircuit;
DSSText.Command =   'ClearAll';             % Clears all instances of OpenDSS-PM
DSSText.Command =   'Set Parallel=Yes';             % Clears all instances of OpenDSS-PM

DSSParallel     =   DSSCircuit.Parallel;    % Habdler for Parallel processing functions
CPUs            =   DSSParallel.NumCPUs;    % Gets how many CPUs this PC has
% By default one actor is created by default, if you want more than one
% parallel instance you will have to create them. Try to leave at least
% One CPU available to handle the rest of windows, otherwise will block
% Everything
disp('Creating Actors');
for i=1:CPUs-1,
    if i ~= 1,
        DSSParallel.CreateActor; % Creates additonal actors
    end;
    DSSText.Command =   'compile (C:\Program Files\OpenDSS\EPRITestCircuits\ckt5\Master_ckt5.DSS)';
    DSSCircuit.Solution.Solve;
    DSSParallel.Wait;   % for the first solve, it is needed to wait before creating other actor
    DSSText.Command =   'set mode=Time stepsize=1h number=16000';
end;
% Now the actors are solved
disp('Simulation Started');
DSSCircuit.Solution.SolveAll;
```

Parallel processing in OpenDSS using COM

Example # 3 MATLAB

```
26  pause(0.1);
27  hold on;
28  BoolStatus      =   0;
29  while BoolStatus == 0,
30      ActorStatus     =   DSSParallel.ActorStatus;
31      BoolStatus      =   all(ActorStatus & 1); %Checks if everybody has ended
32      ActorProgress   =   DSSParallel.ActorProgress;
33      bar(ActorProgress);
34      axis([0 (CPUs) 0 100]);
35      xlabel('Actor #');
36      ylabel('Actor progress (%)');
37      pause(0.5);  %  A little wait to not saturate the Processor
38  end;
39  disp('Simulation finished by all the actors');
```

Complementary Tools

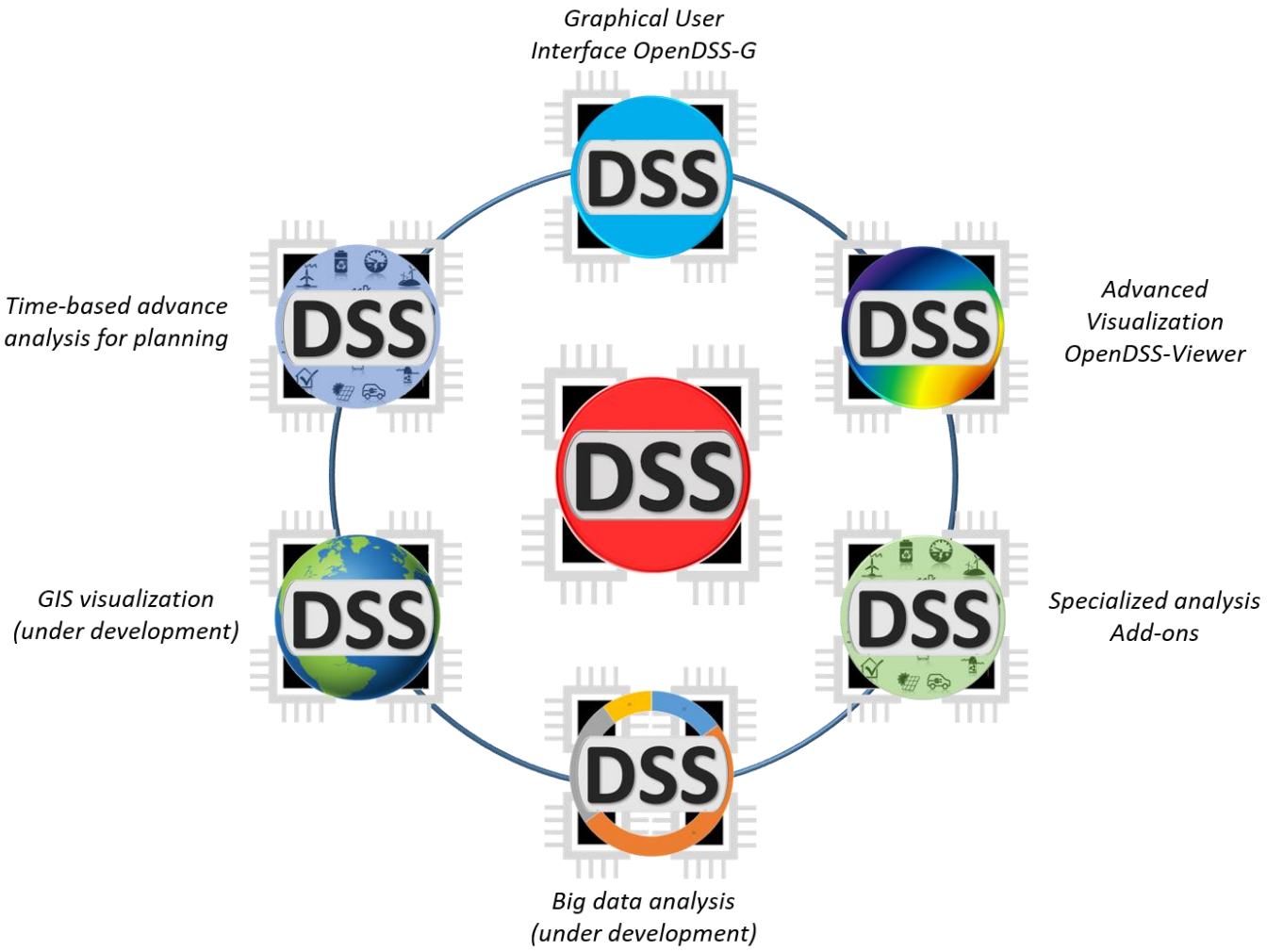
DEMOS

Introduction to OpenDSS-G

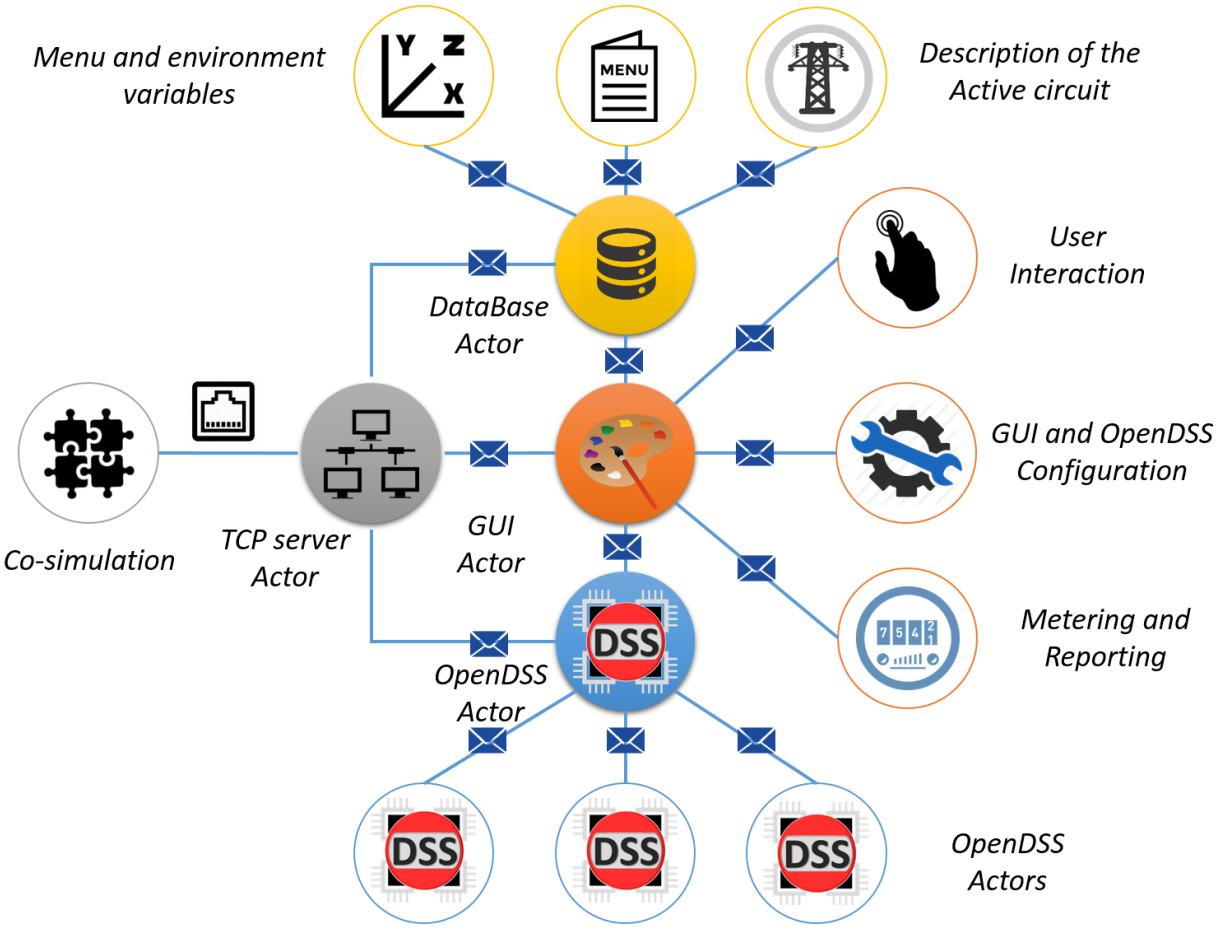
OpenDSS-G YouTube channel

https://www.youtube.com/channel/UCGe58SDH3Iq-EGvnxEOuWaQ

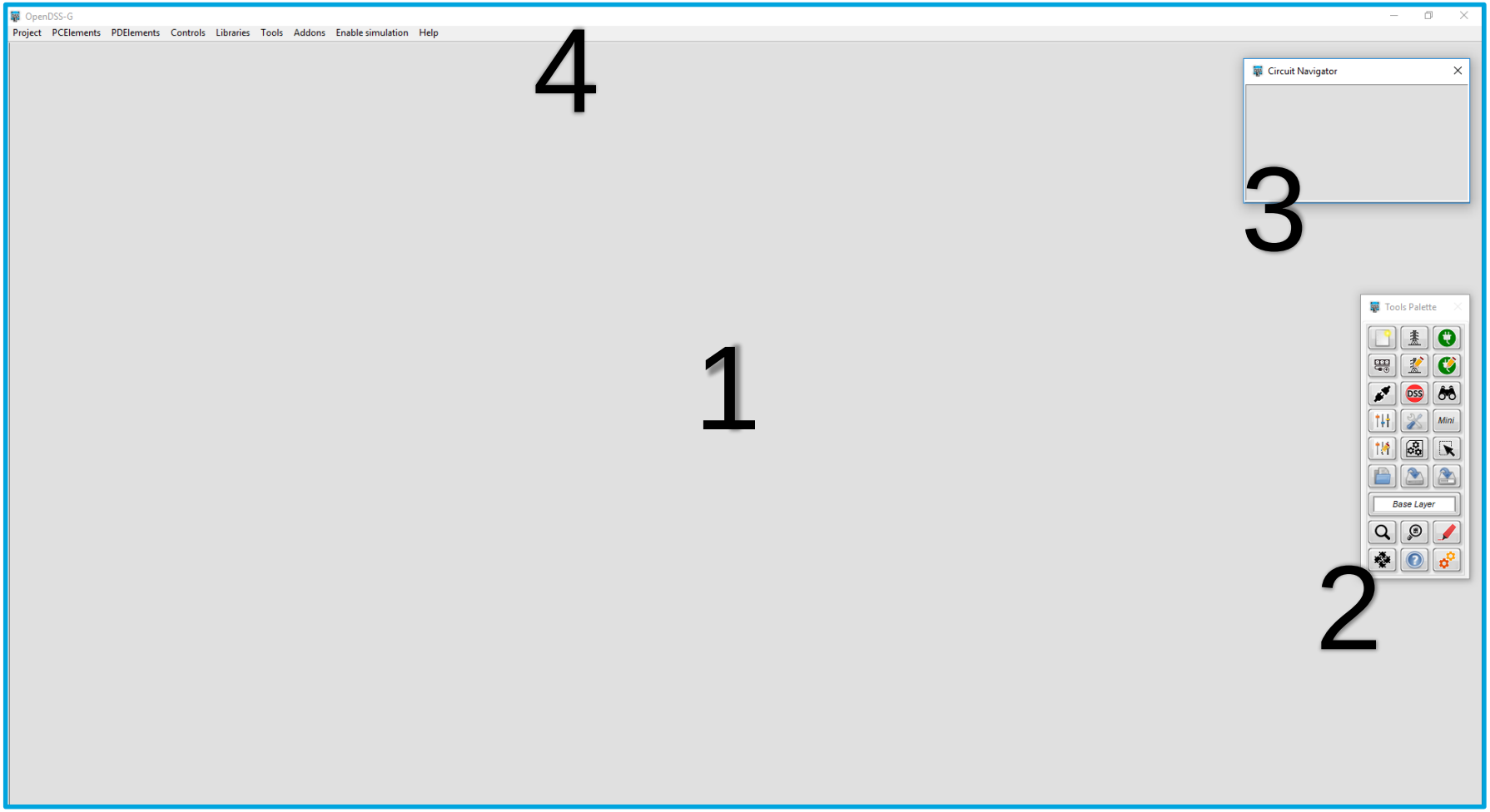
OpenDSS derivative products



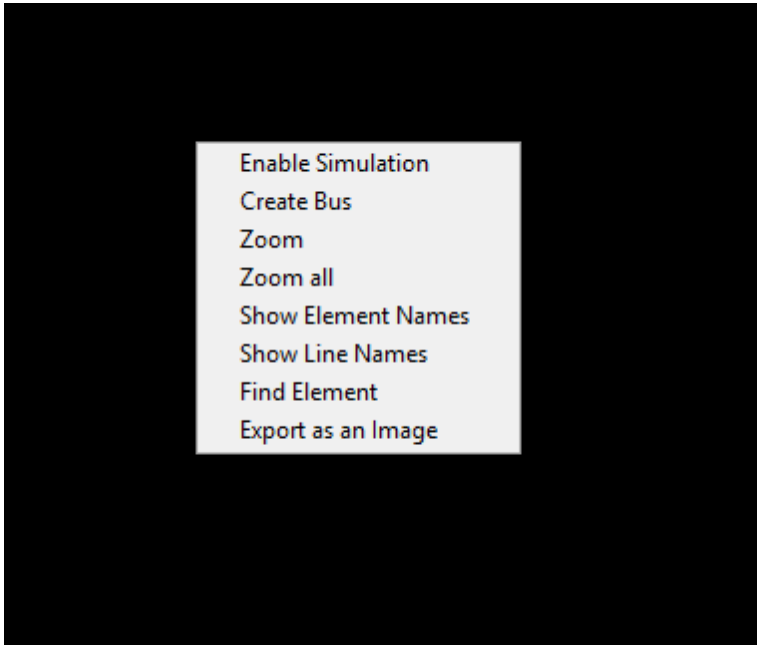
Introduction to OpenDSS-G

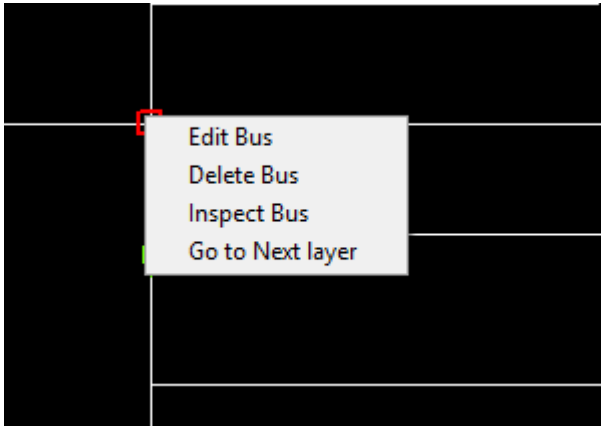


Introduction to OpenDSS-G

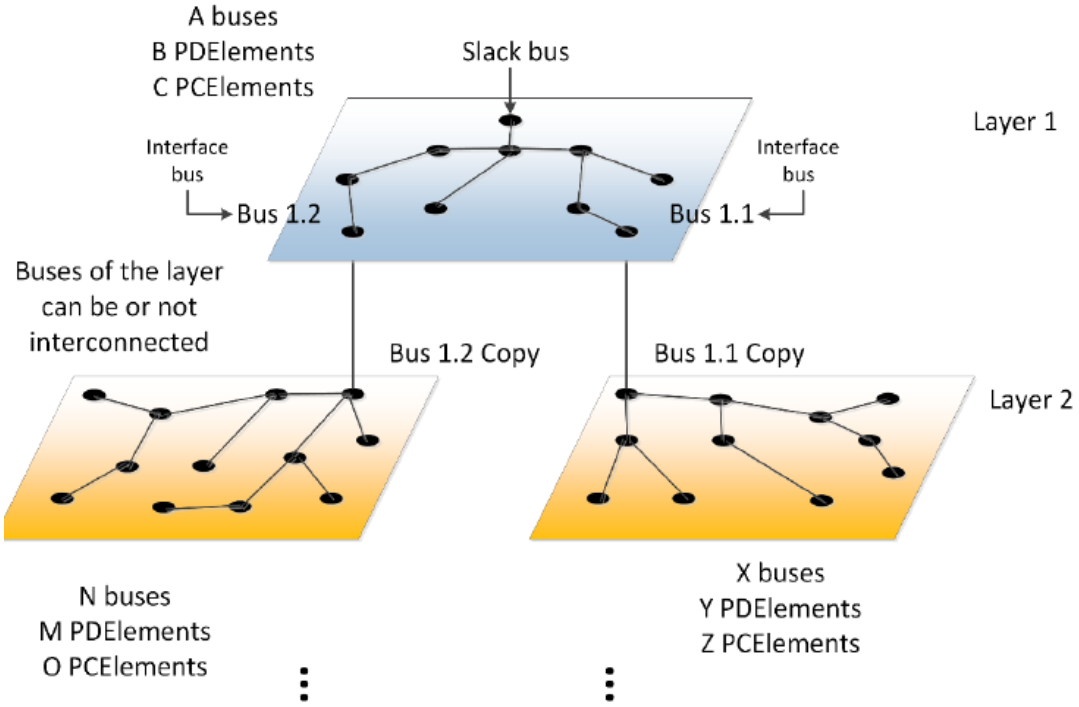


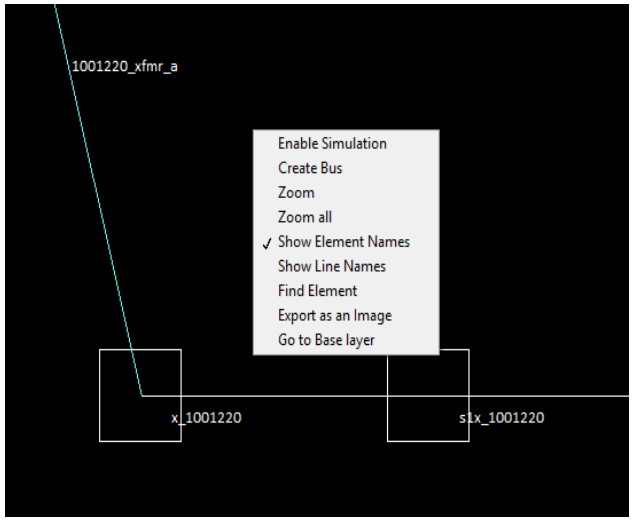
Introduction to OpenDSS-G

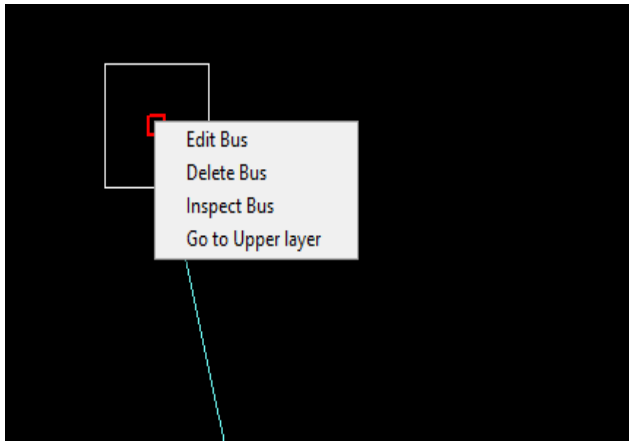




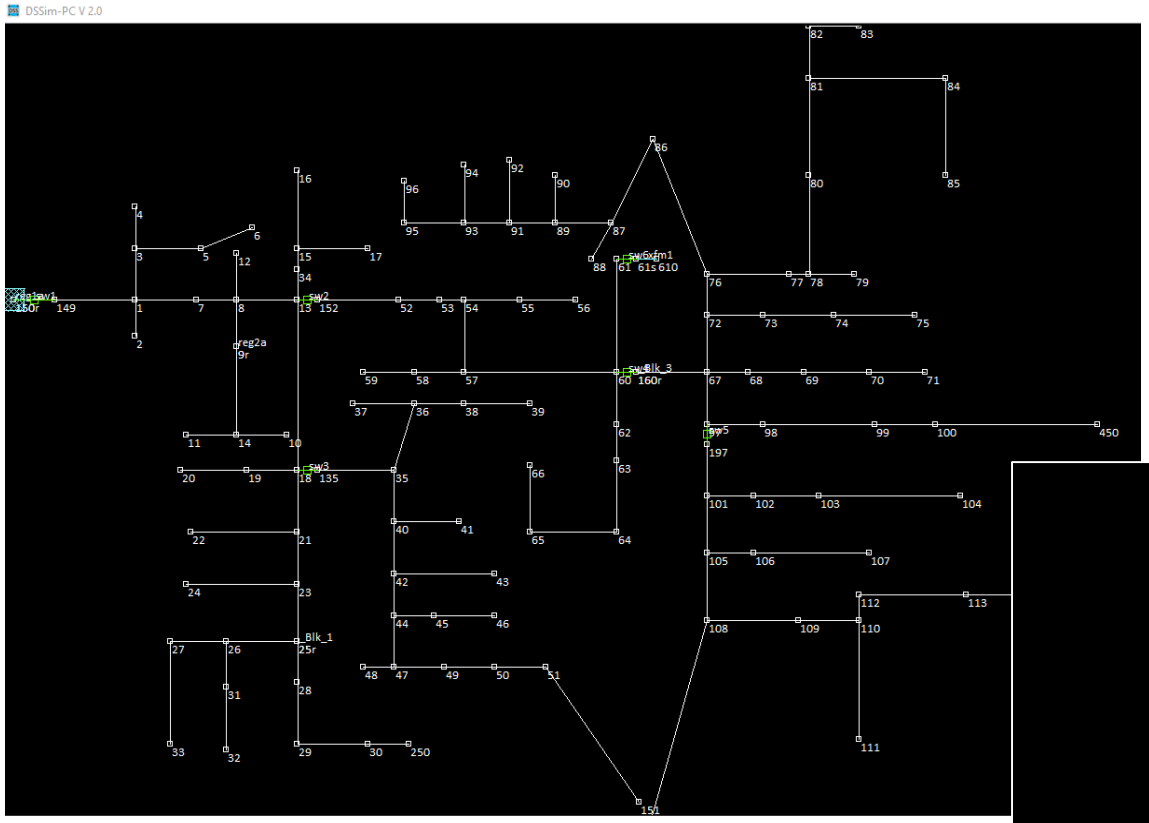
Introduction to OpenDSS-G





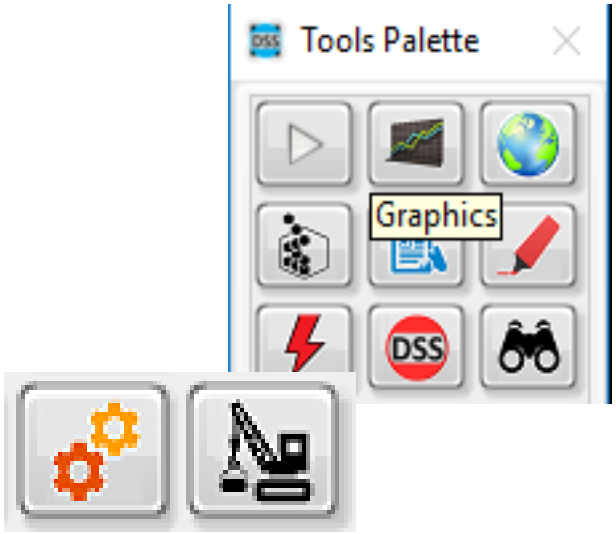


Introduction to OpenDSS-G

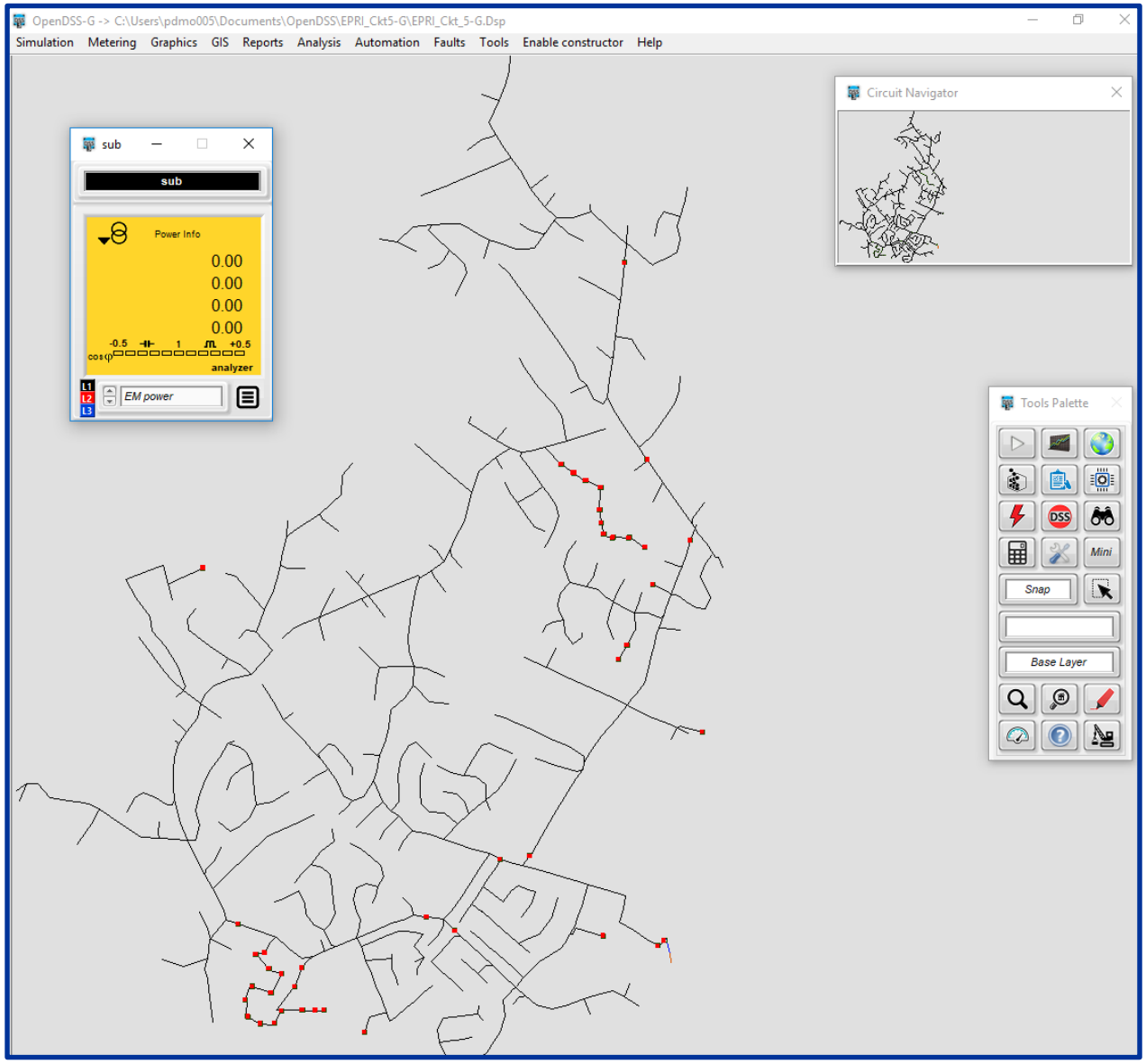


Introduction to OpenDSS-G

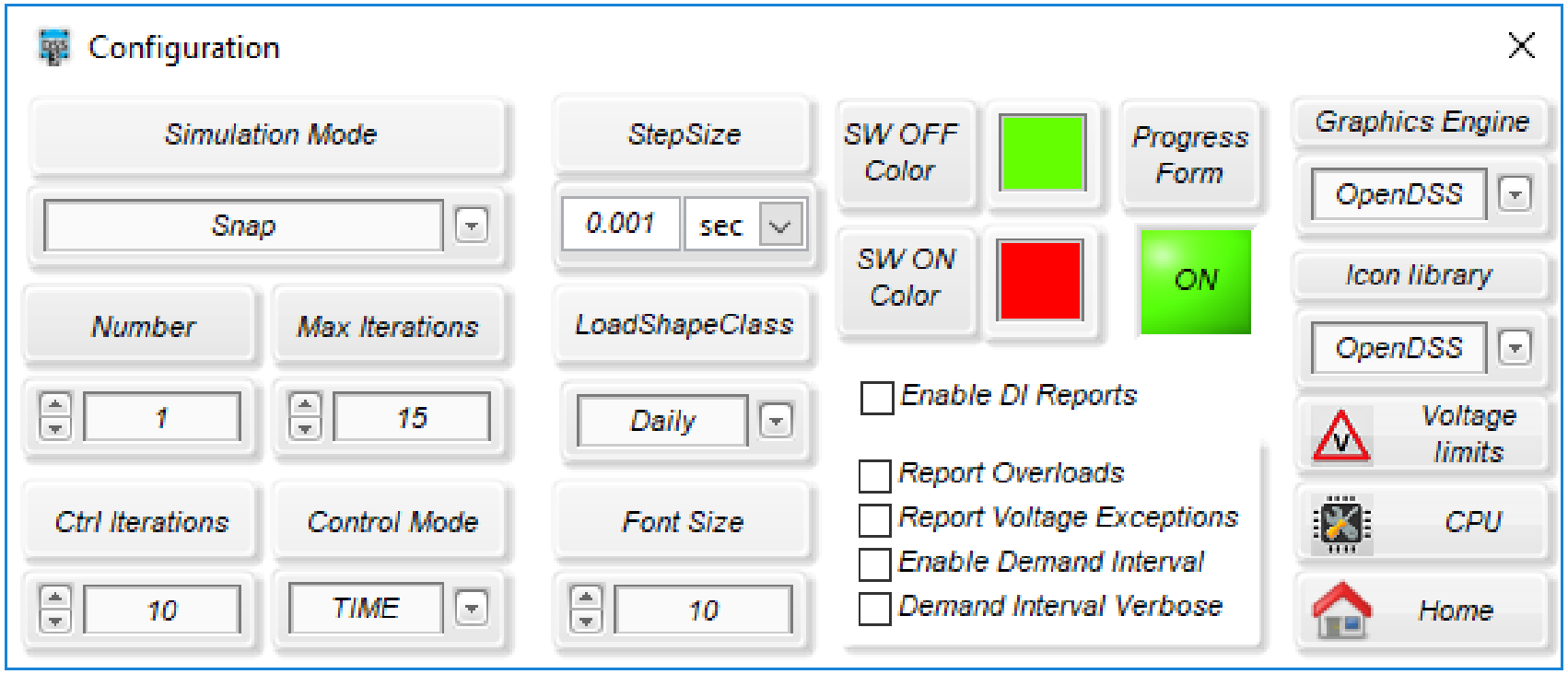
Introduction to OpenDSS-G



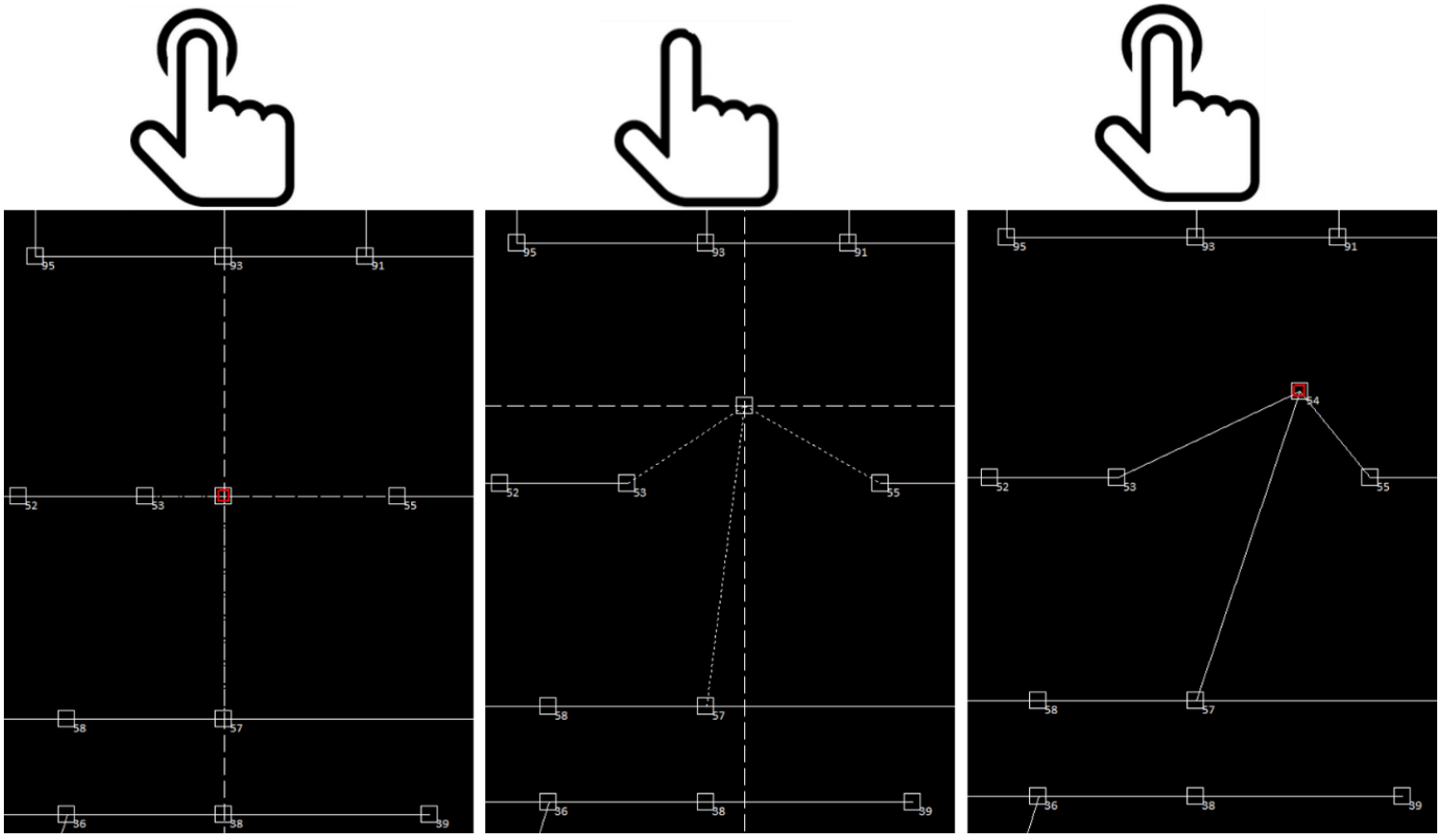
Introduction to OpenDSS-G



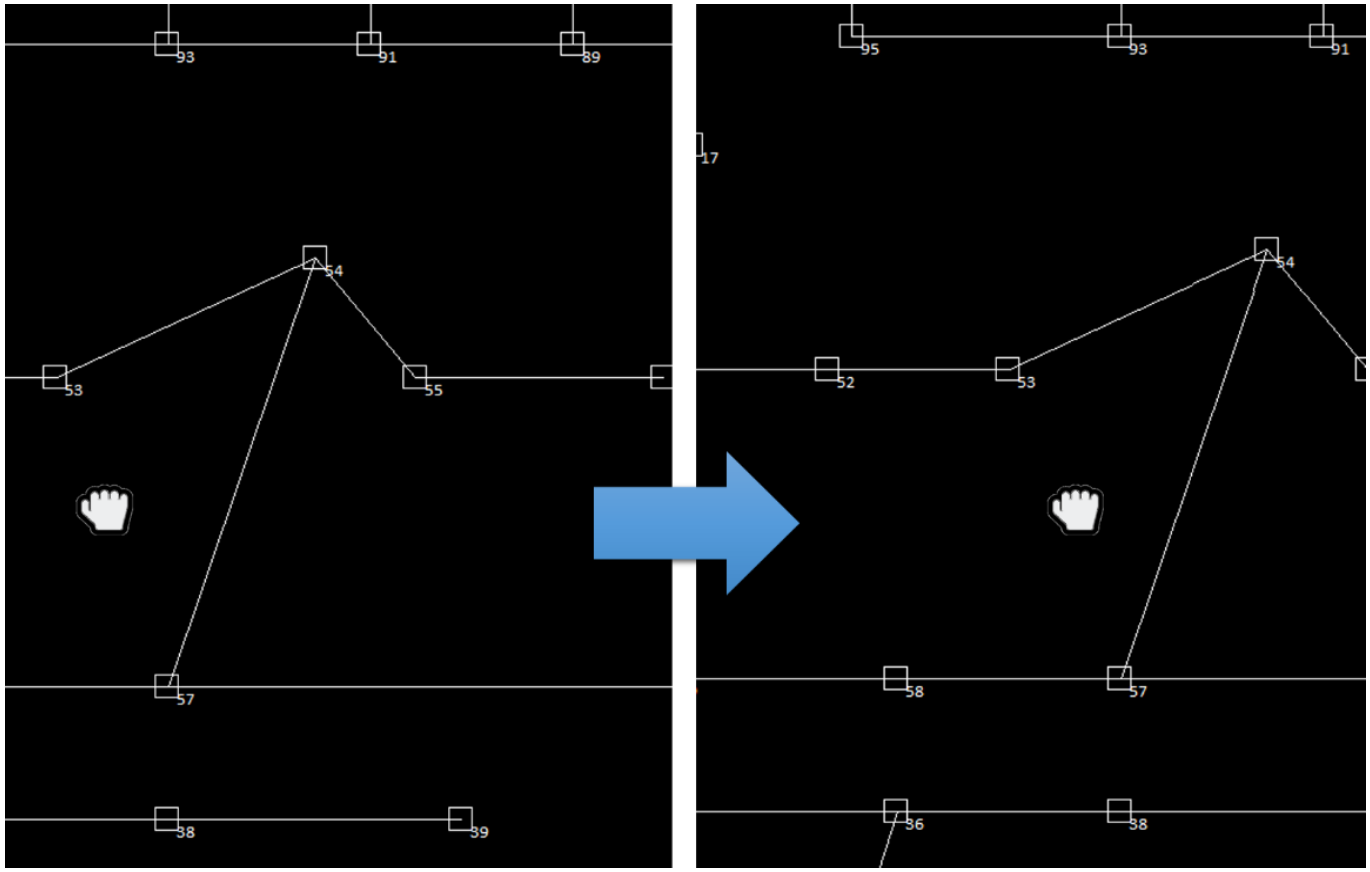
Introduction to OpenDSS-G



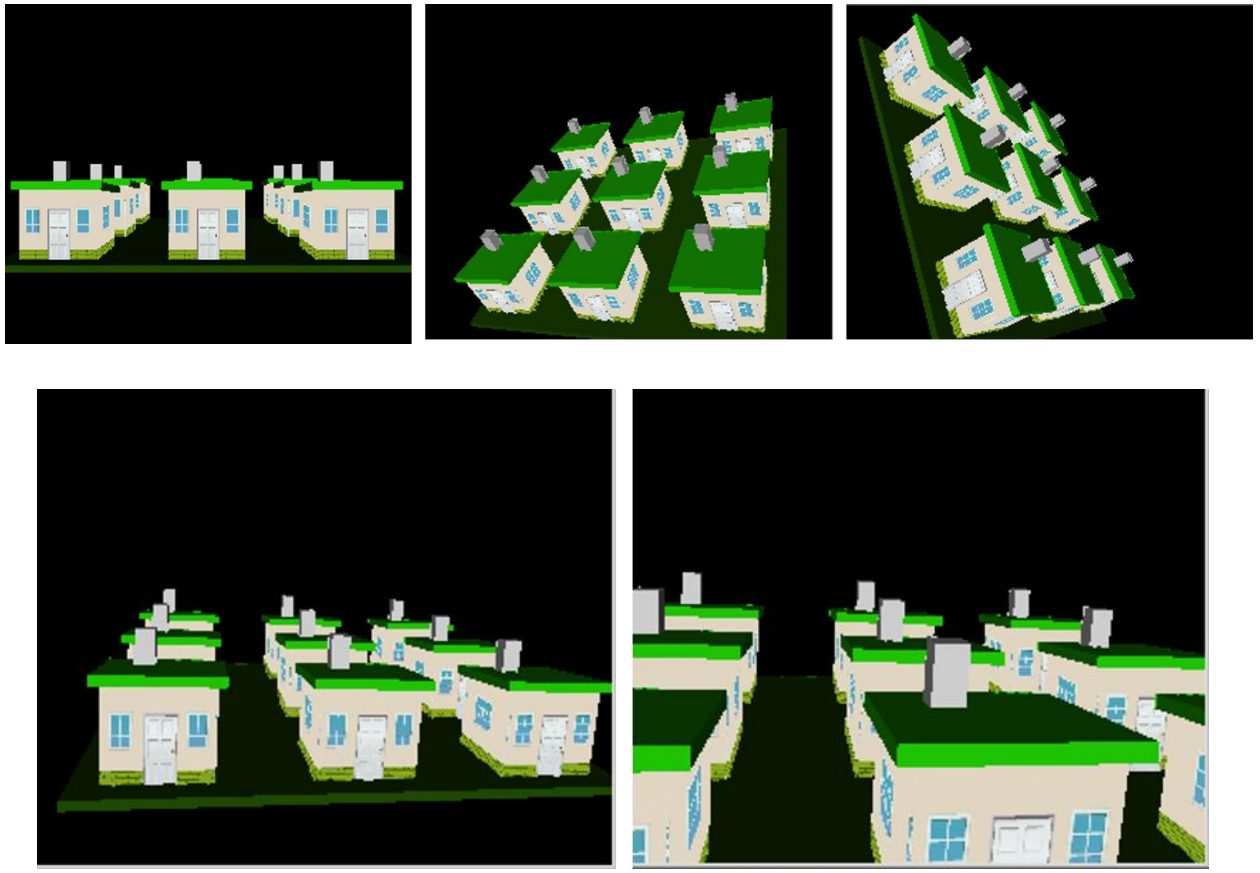
Introduction to OpenDSS-G



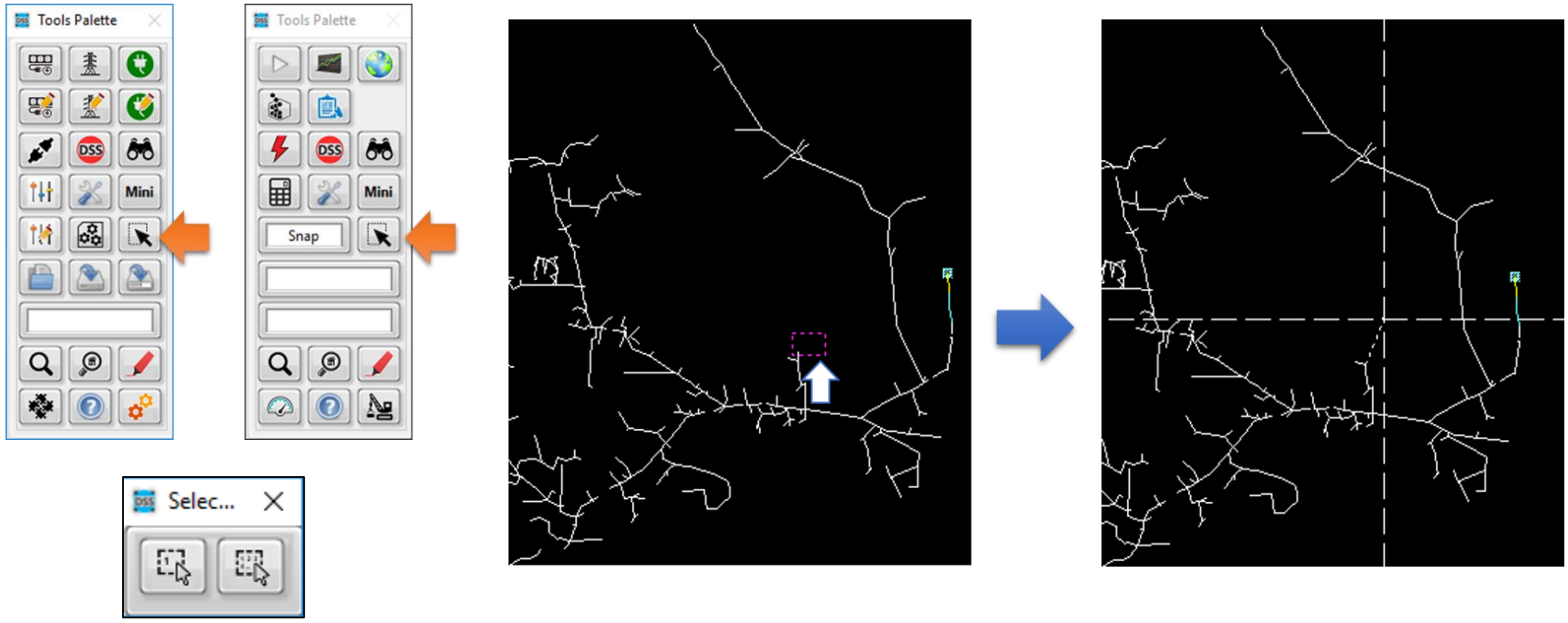
Introduction to OpenDSS-G



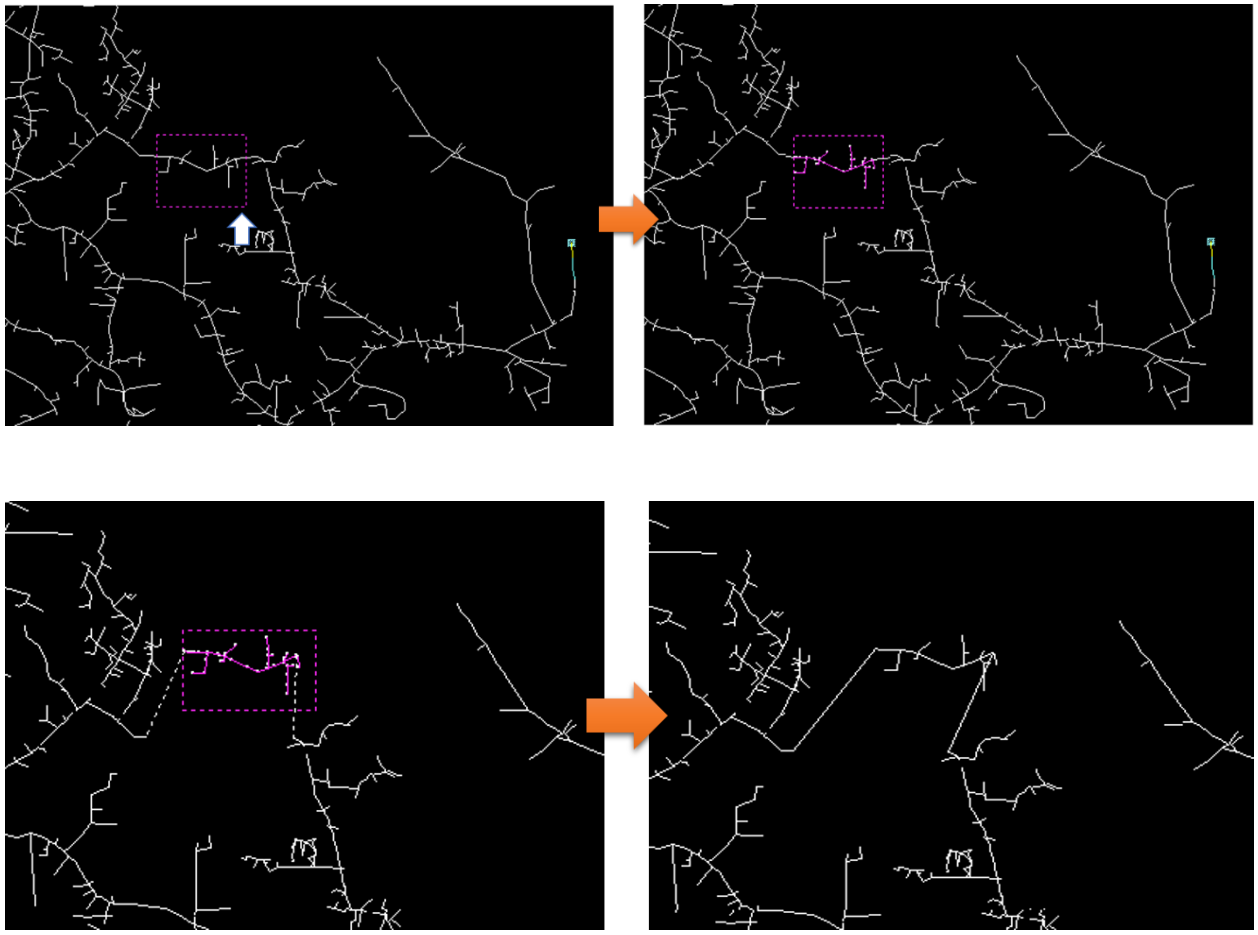
Introduction to OpenDSS-G



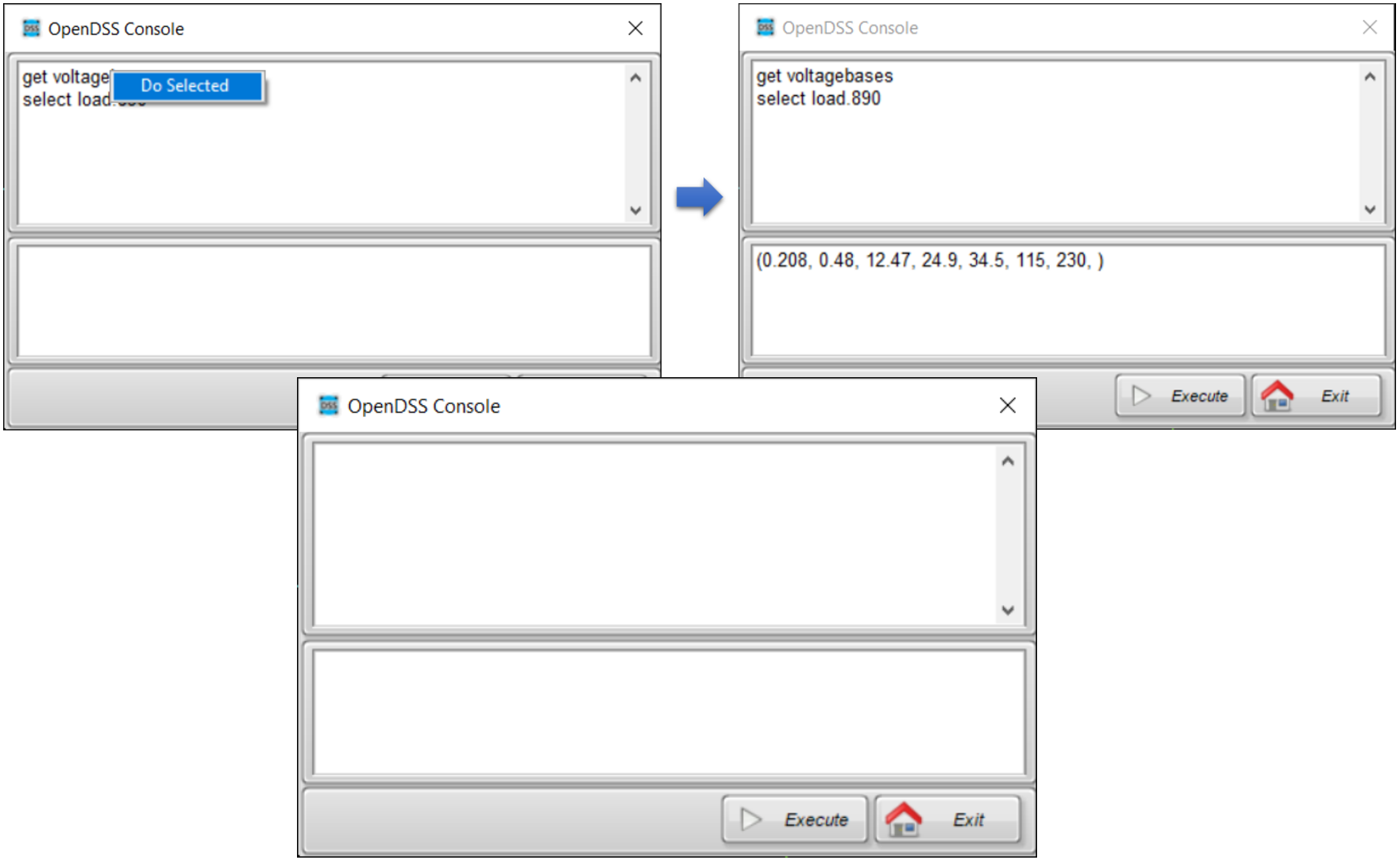
Introduction to OpenDSS-G



Introduction to OpenDSS-G



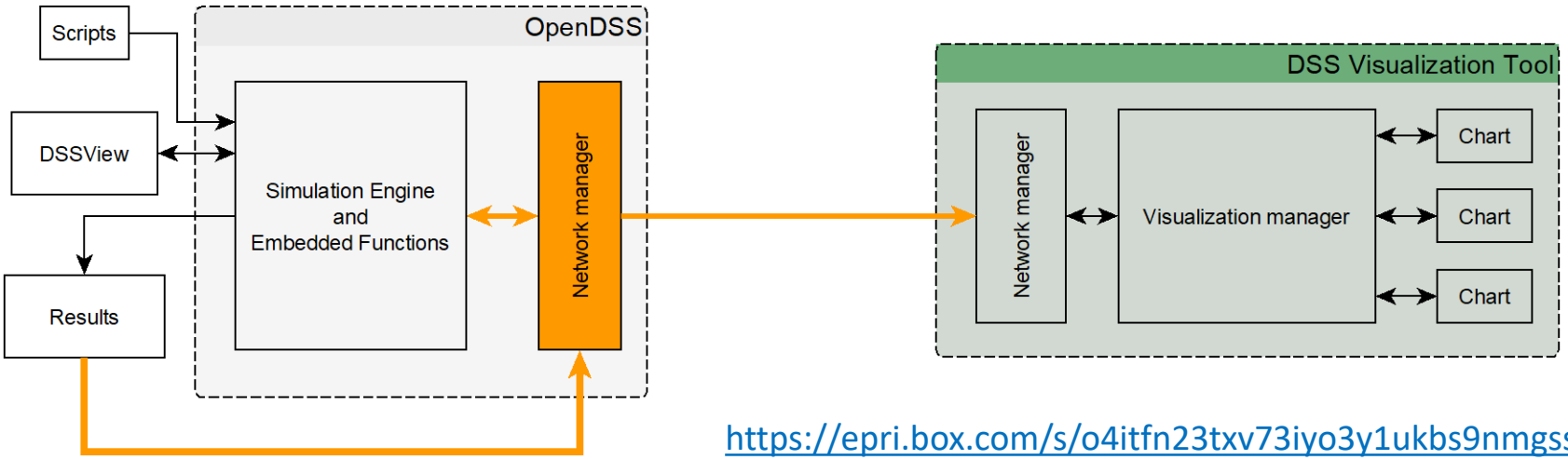
Introduction to OpenDSS-G



Complementary Tools

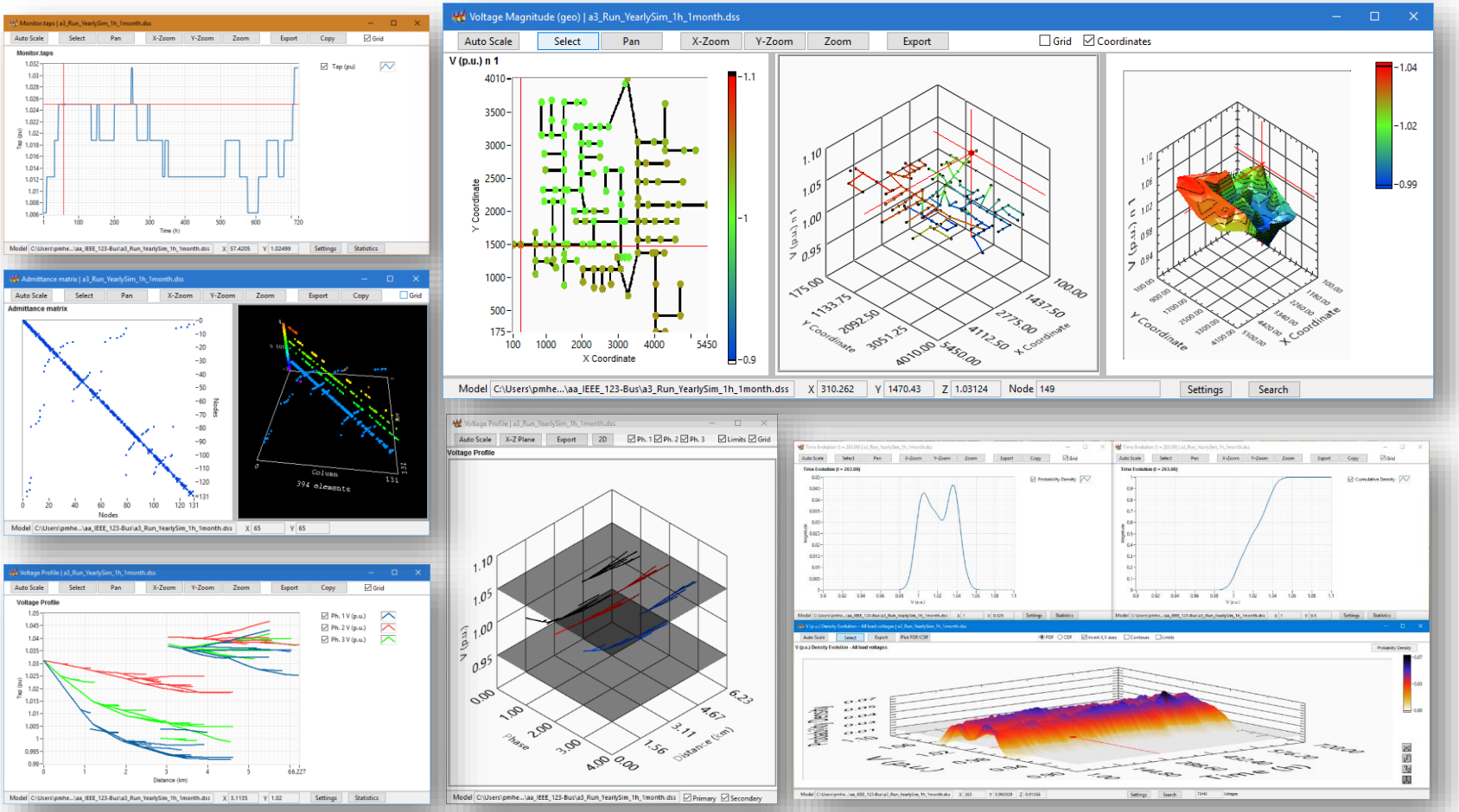
Advanced Graphics Module for OpenDSS (OpenDSS-Viewer)

Developed by Miguel Hernandez (EPRI). Enhance the visualization of Distribution System Simulations with a **flexible**, **scalable** and **meaningful** approach.



https://epri.box.com/s/o4itfn23txv73iyo3y1ukbs9nmgsspgo

Complementary Tools



Importing/creating models

Creating your model

You have 2 options:

1. Importing an existing model

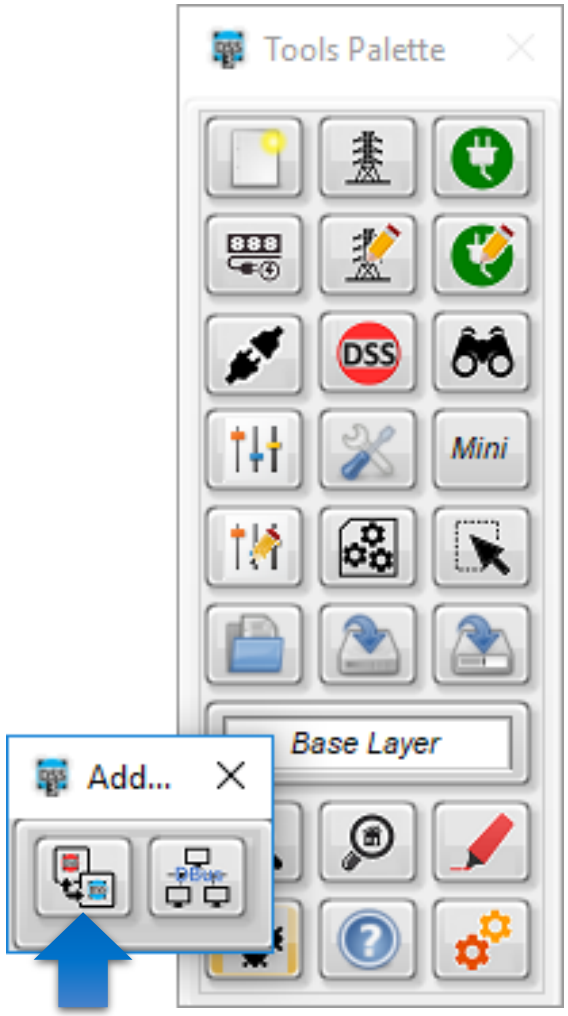
2. Describing your circuit from the scratch

Before anything

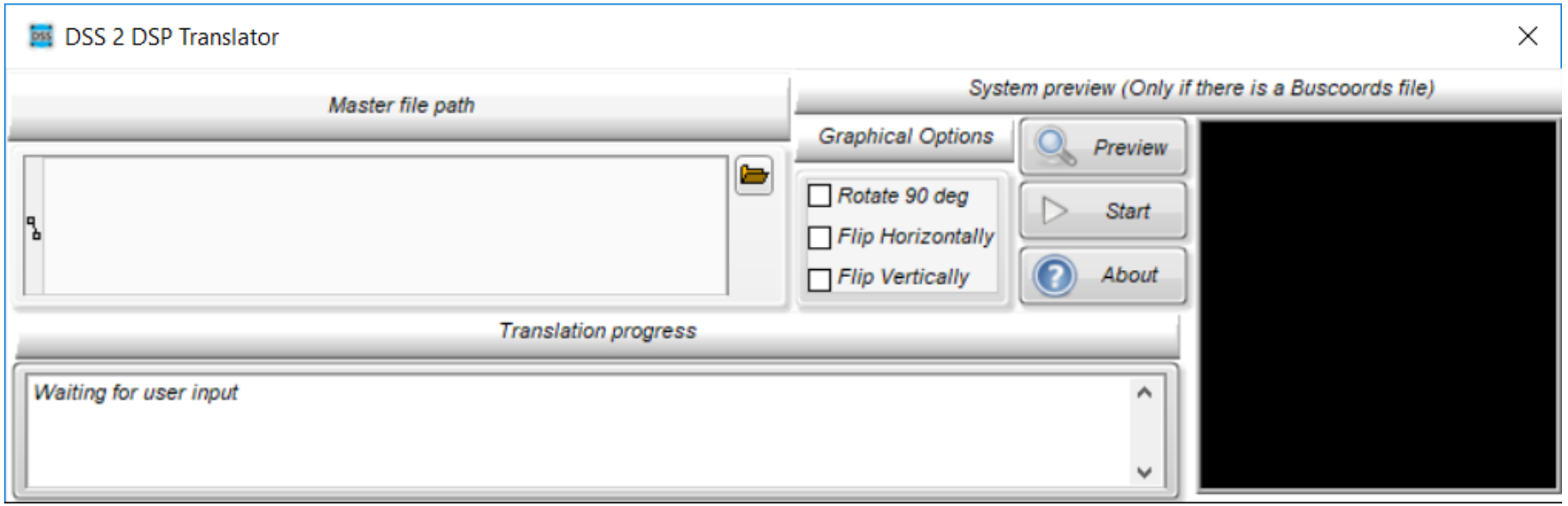
Download the examples to be used in this session:

https://sourceforge.net/p/dssimpc/code/HEAD/tree/trunk/Distribution/Examples/

Importing a model

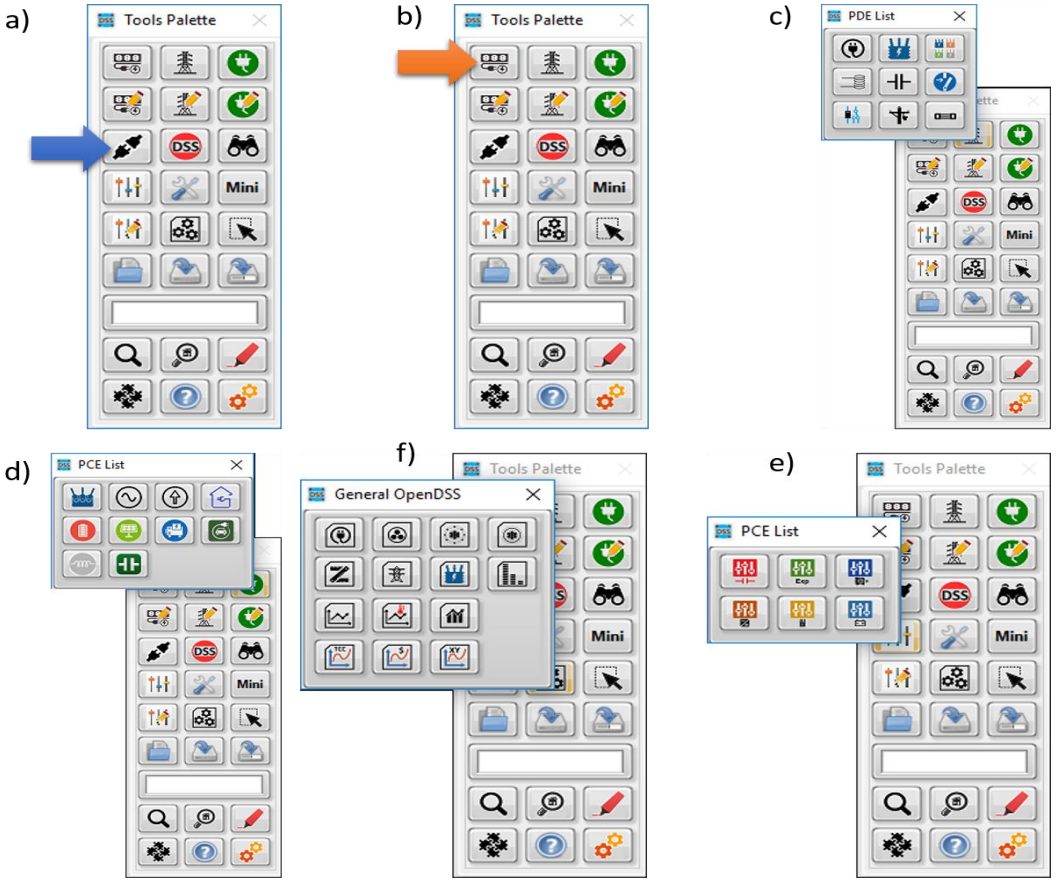


Importing a model

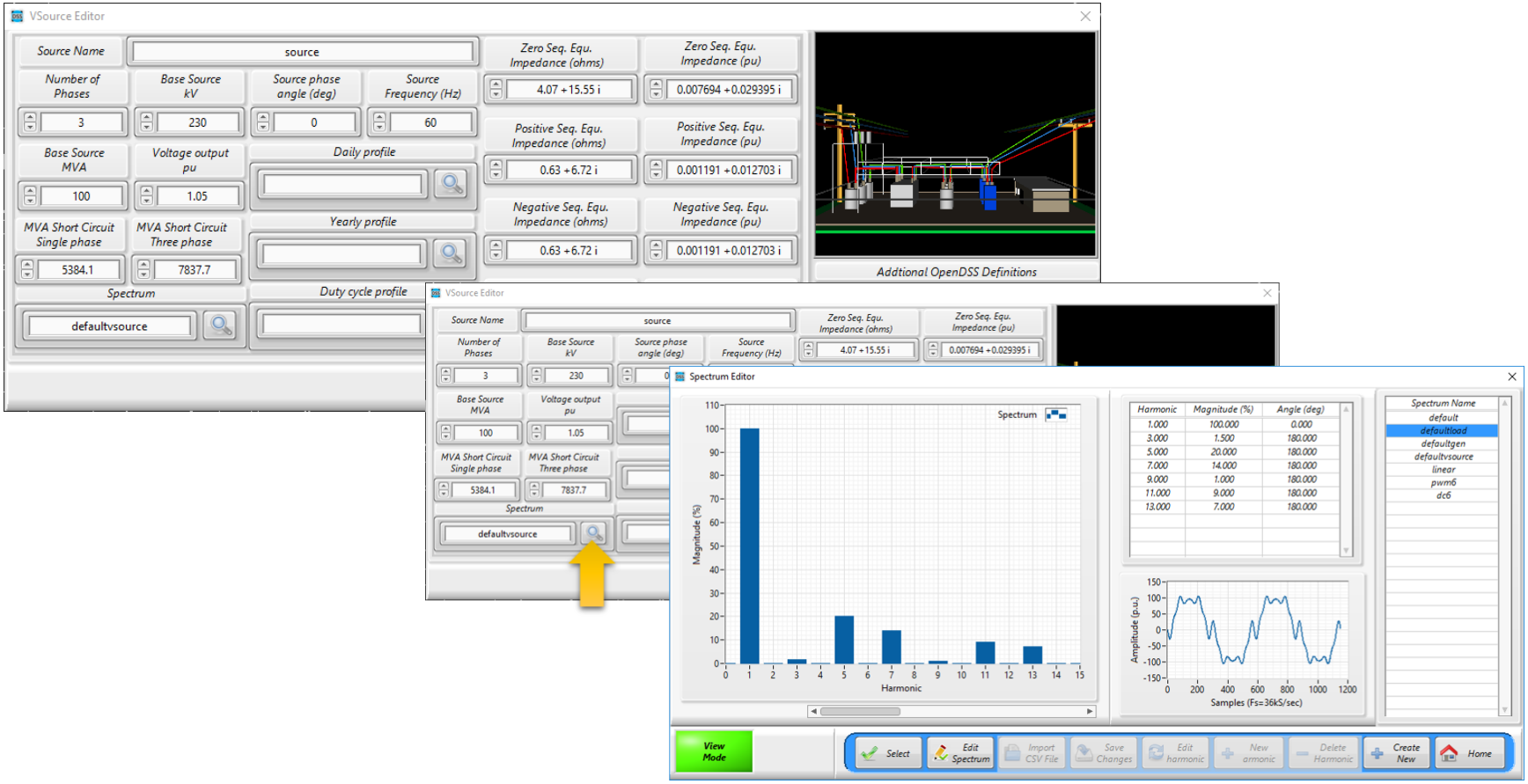


Import demo

Creating the model from the scratch



Creating the model from the scratch



Creating the model from the scratch

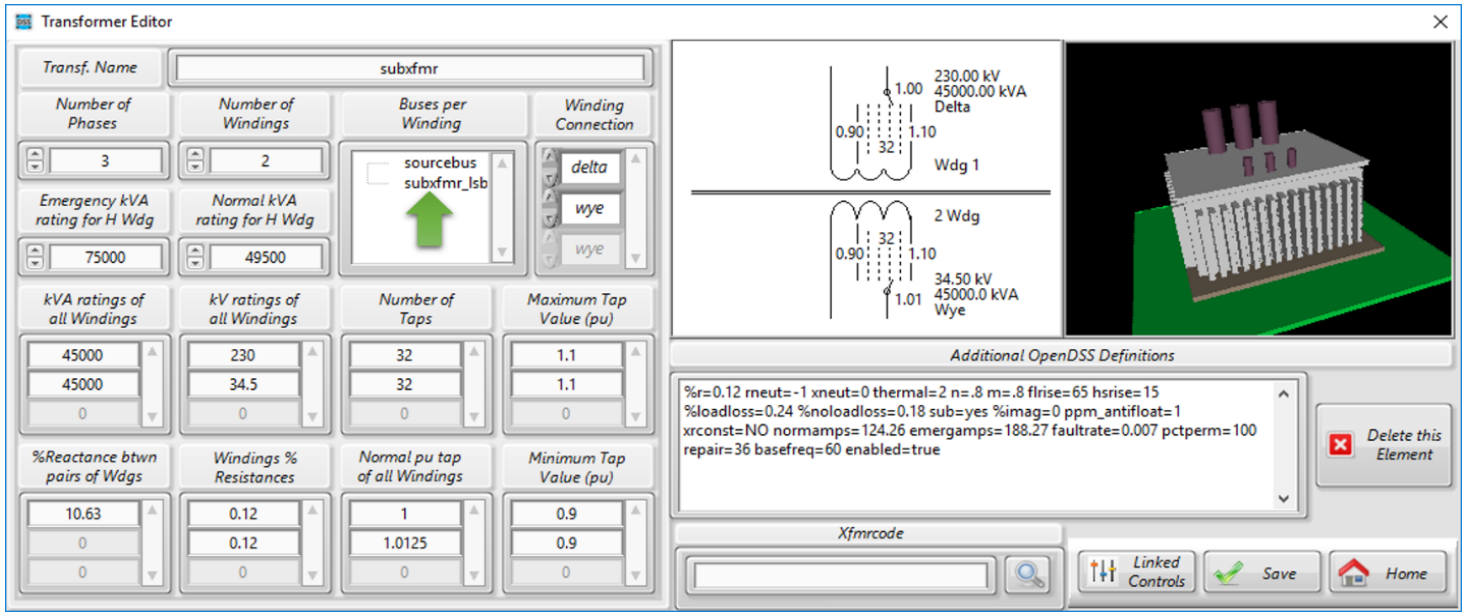
Rules:

1. Connect your PD/PCElement to an existing bus

2. Follow the instructions

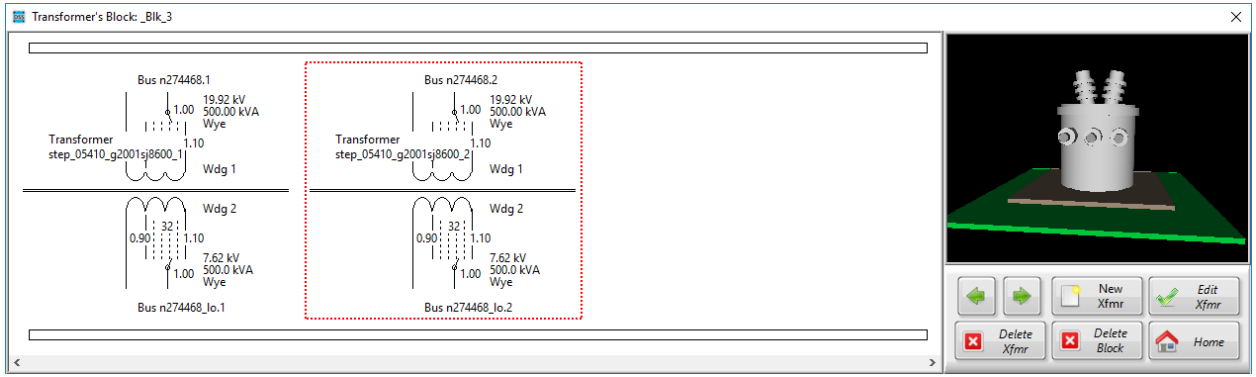
Creating the model from the scratch

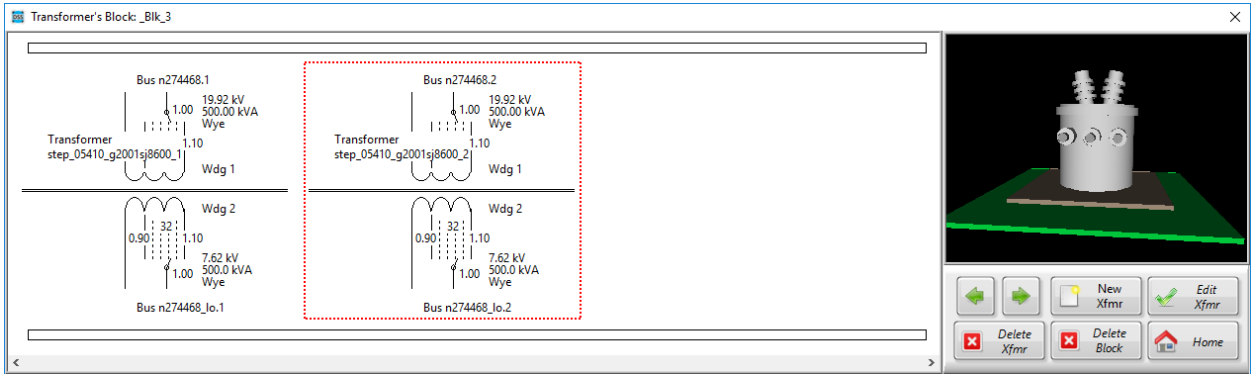
Transformers (special case):



Creating the model from the scratch

Transformers (special case):





Creating model from the scratch demo

Using OpenDSS-G (Simulation)

LET'S SIMULATE

Using OpenDSS-G (Simulation)

Today's examples:

1. Using Storage devices
2. Yearly Simulations and parallel processing
3. Real-time Simulation
    - Checking regulators and protections coordination
    - Using the FLISR programmer

Before anything

Download the examples to be used in this session:

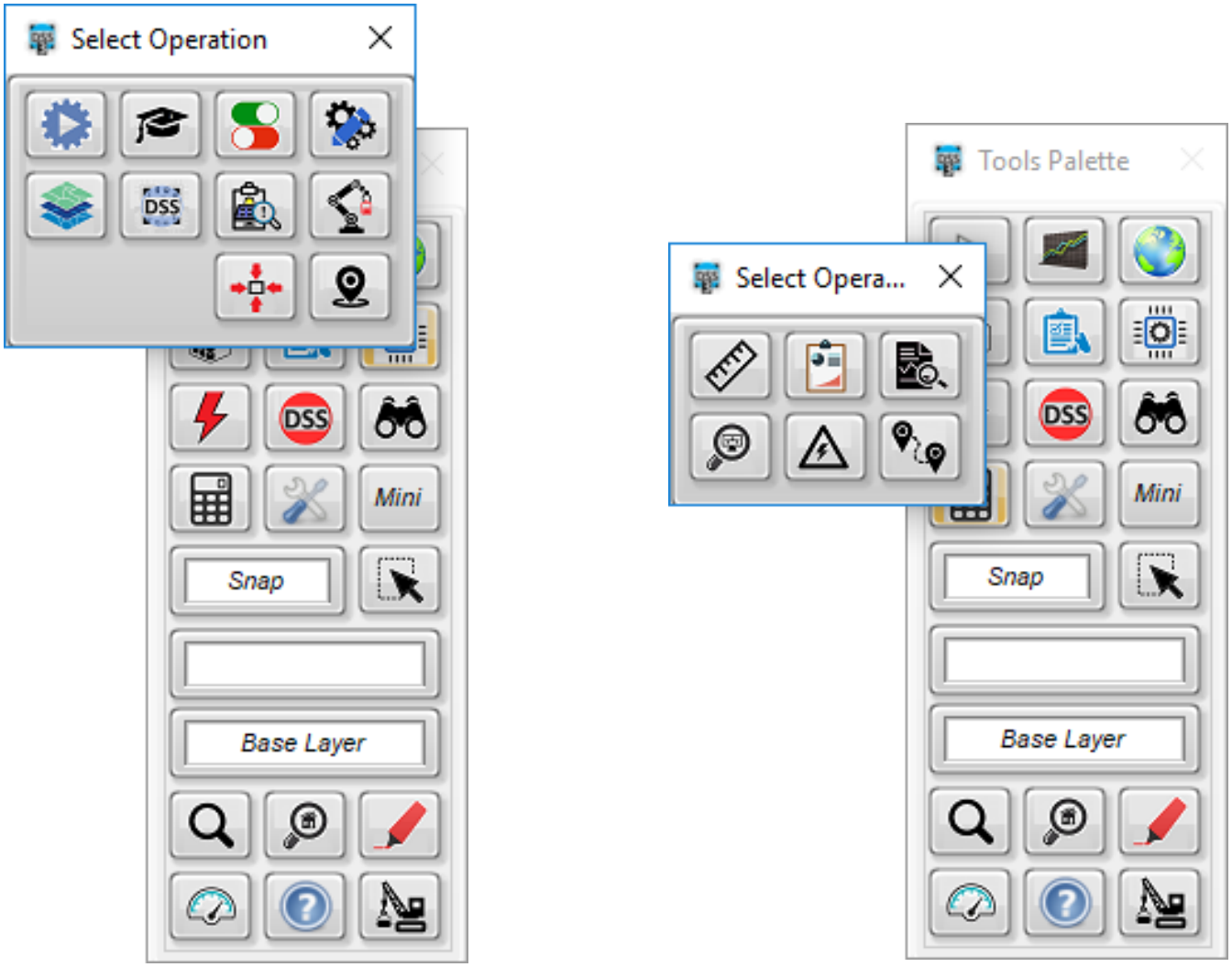
https://sourceforge.net/p/dssimpc/code/HEAD/tree/trunk/Distribution/Examples/

Solving thermal issues with storage devices

Harmonics simulation

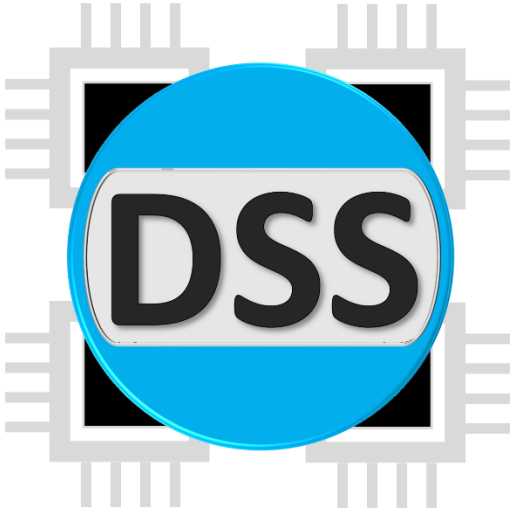
Automation tools

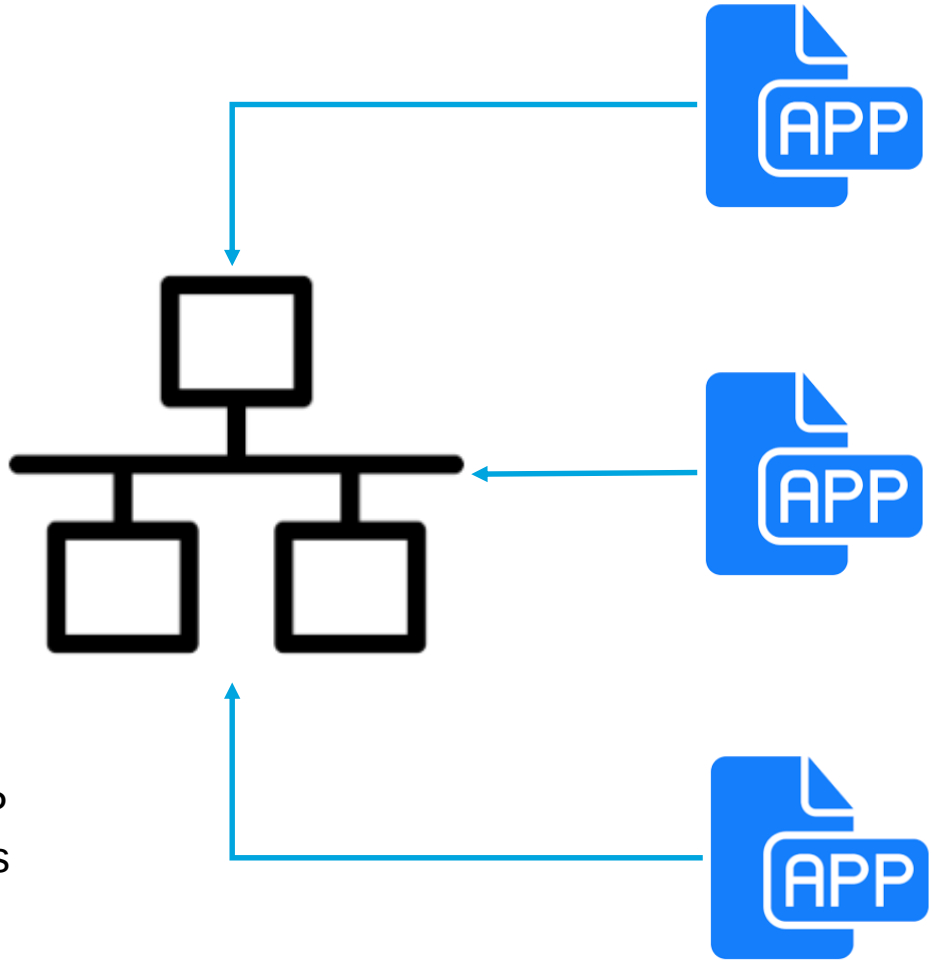
Automation tools



Automation tools demo

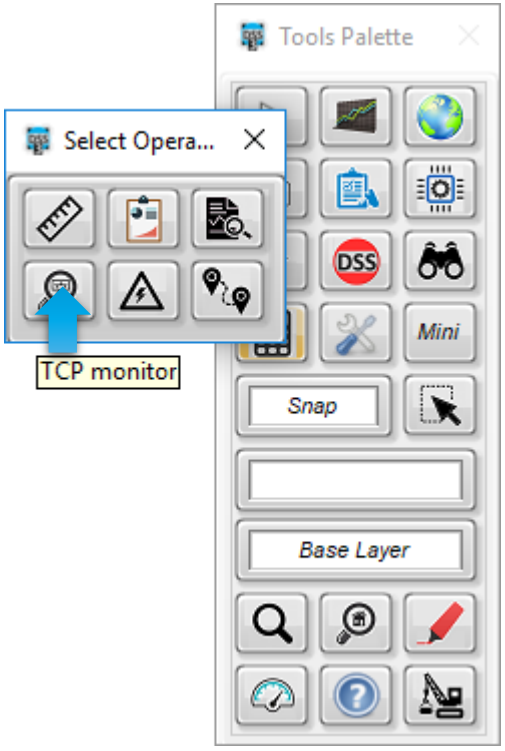
Co-Simulation

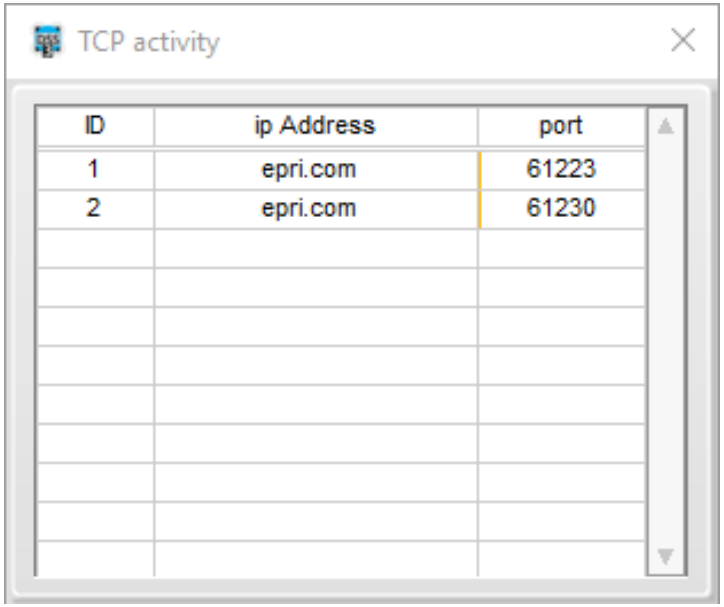




OpenDSS-G Incorporates a TCP server for distributed applications

Co-Simulation

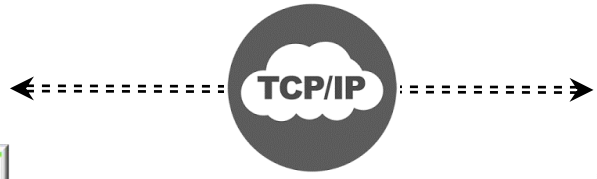


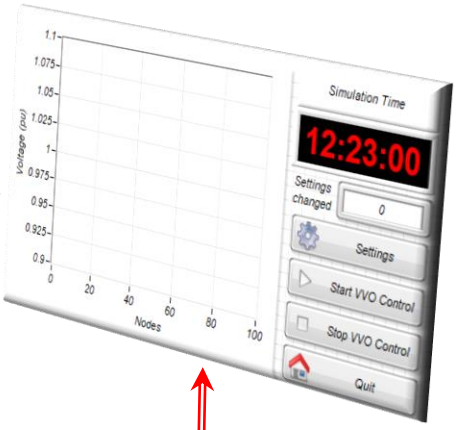


Co-Simulation

VVO control interface

This component handles the TCP /IP communication with OpenDSS-G. It also controls the simulation execution and communicates with the control algorithm written in python.



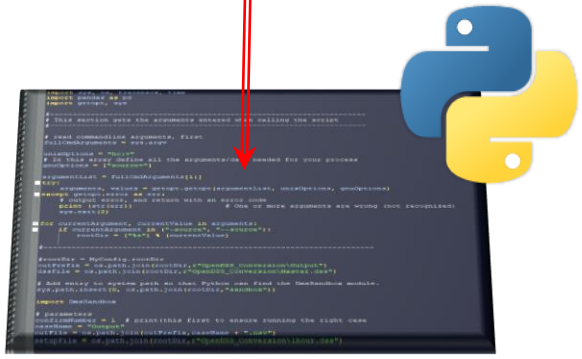


OpenDSS-G

This platform contains the power system model. It performs the QSTS simulation and provides all the services required to remotely control the simulation though the TCP/IP server integrated. This platform also provides the graphical interface and the tools for highlighting system features using color scales, element localization, metering and real-time system's data among others.

The control algorithm

This component evaluates the optimal control settings for a specific hour using a local model of the simulated power system in OpenDSS-G. The output of the algorithm are the control settings for all the controllable devices on the grid.



Co-Simulation

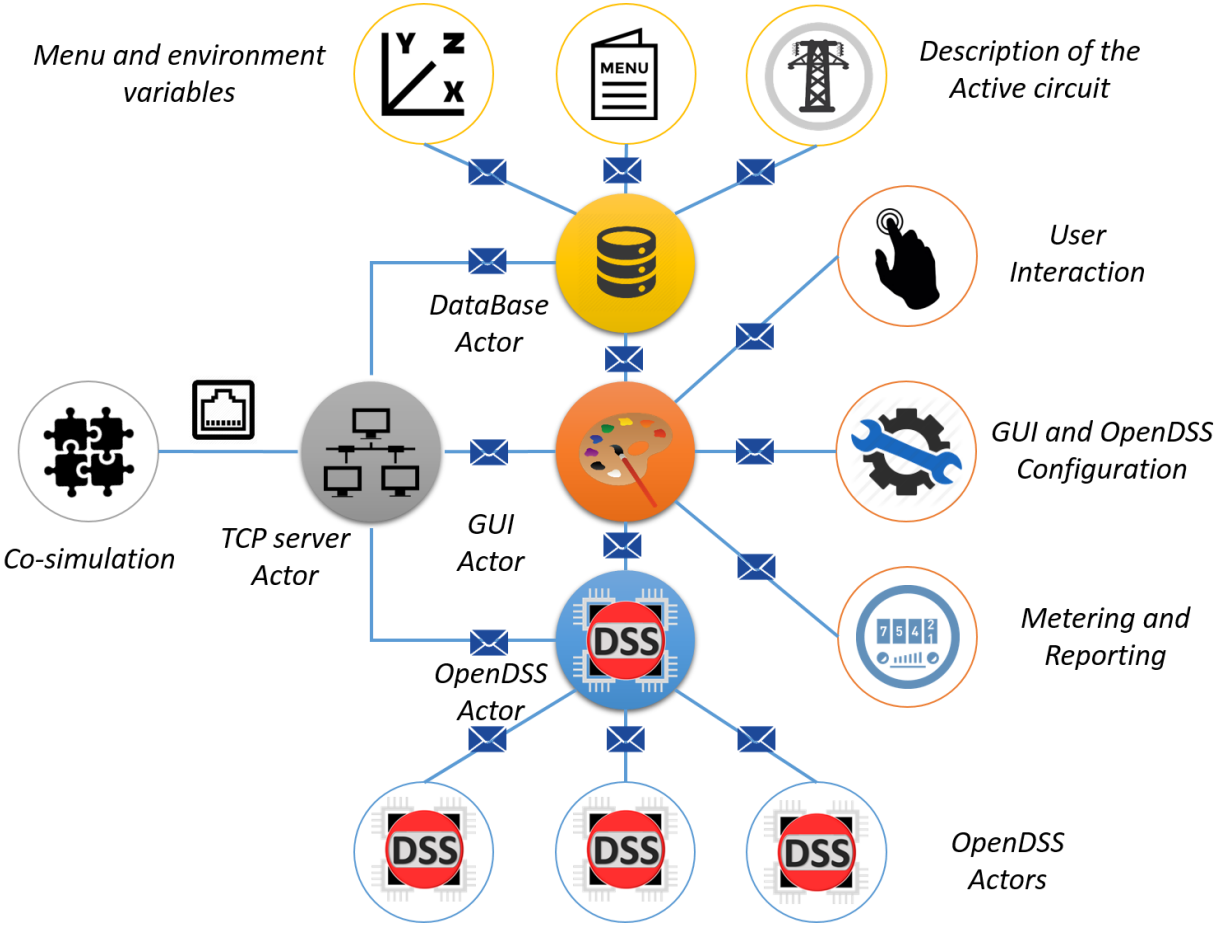
Documentation:

https://sourceforge.net/projects/dssimpc/files/Documents/DSTCP-Comm.pdf/download

https://sourceforge.net/projects/dssimpc/files/Documents/MATLAB_Lib.pdf/download

Co-simulation example

Changing the language of your local installation



Changing the language of your local installation

Documentation:

https://sourceforge.net/projects/dssimpc/files/Documents/Changing_the_language_of_OpenDSS-G.pdf/download

https://sourceforge.net/projects/dssimpc/files/Documents/Changing_the_language_of_OpenDSS-G-menus.pdf/download

Questions ??

Together...Shaping the Future of Electricity