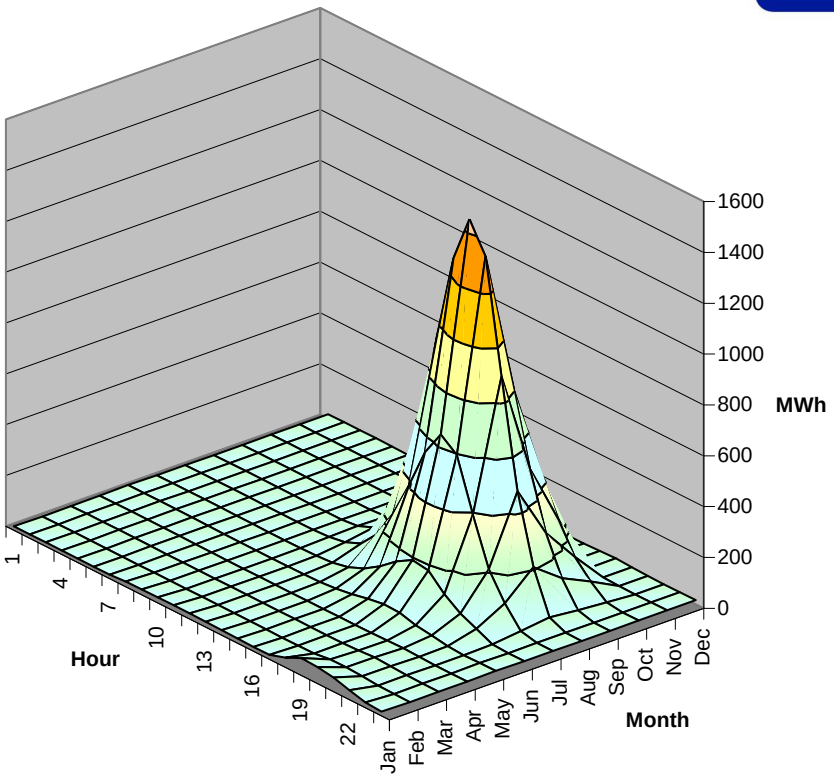



ELECTRIC POWER RESEARCH INSTITUTE

# OpenDSS Introductory Training Level 1

28 APR 2009

Roger Dugan
rdugan@epri.com

EPRI | ELECTRIC POWER RESEARCH INSTITUTE

# Introduction

# Why DSS?

- DSS was developed to provide a very flexible research platform and a foundation for special distribution analysis applications such as DG analysis
- Fills gaps left by other distribution system analysis tools.
  - These do very well in traditional distribution system analysis meeting the needs of their respective user bases
  - Integration of user interface, GIS and other databases quite important, but results in slower implementation of innovative modeling to meet new challenges

# Current Related EPRI Activities

- Intelligrid
  - Distribution Fast Simulation & Modeling
  - DSE – Distribution State Estimator
- CIM/DCIM
- OpenDSS – Distribution System Simulator
  - Multipurpose distribution system analysis tool
  - Open source version has been released – 5 Sept 2008
  - Official release – November 2008
  - (Focus of this Presentation)

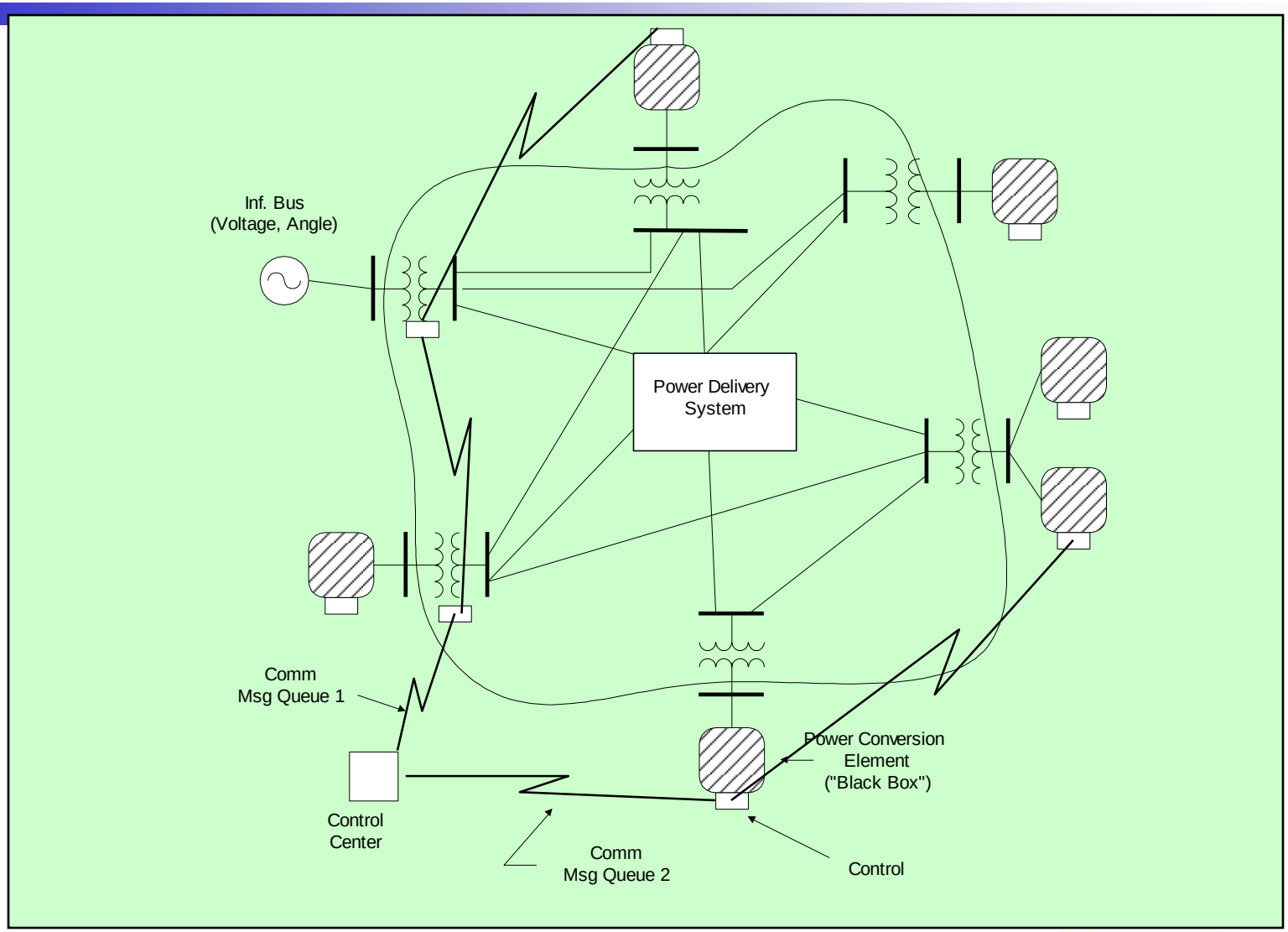
# DSS Background

- Under development for more than 10 Years
  - Started at Electrotek Concepts in 1997
  - Purchased by EPRI in 2004
- Objectives in 1997
  - Tool to support all distribution planning aspects of distributed generation
  - Implement a flexible research platform
  - Incorporate object-oriented data ideas
- Key Future work
  - Platform for DSE for North American Systems
  - Research platform for reliability tools

# Distribution System Simulator (DSS)

- The DSS is designed to simulate utility distribution systems in arbitrary detail for most types of analyses related to distribution planning.
  - It performs its analysis types in the *frequency domain*,
    - Power flow,
    - Harmonics, and
    - Dynamics.
  - It does NOT perform electromagnetic transients (*time domain*) studies.

# Overall Model Concept



# Example DSS Applications

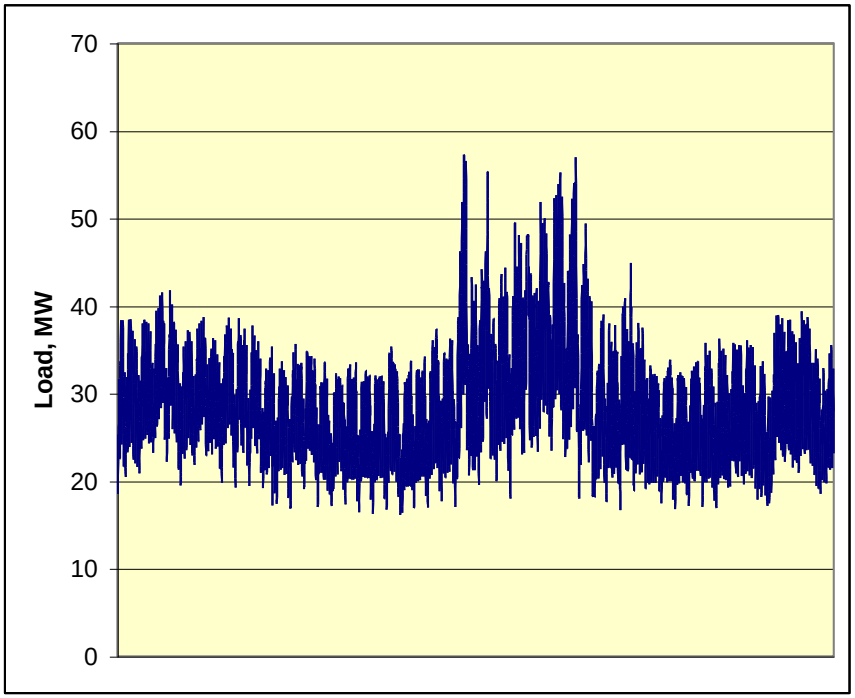
- Neutral-to-earth (stray) voltage simulations.
- Loss evaluations due to unbalanced loading.
- Development of DG models for the IEEE Radial Test Feeders.
- High-frequency harmonic and interharmonic interference.
- Losses, impedance, and circulating currents in unusual transformer bank configurations.
- Transformer frequency response analysis.
- Distribution automation control algorithm assessment.
- Impact of tankless water heaters on flicker and distribution transformers.
- Wind farm collector simulation.
- Wind farm impact on local transmission.
- Wind generation and other DG impact on switched capacitors and voltage regulators.
- Open-conductor fault conditions with a variety of single-phase and three-phase transformer connections.

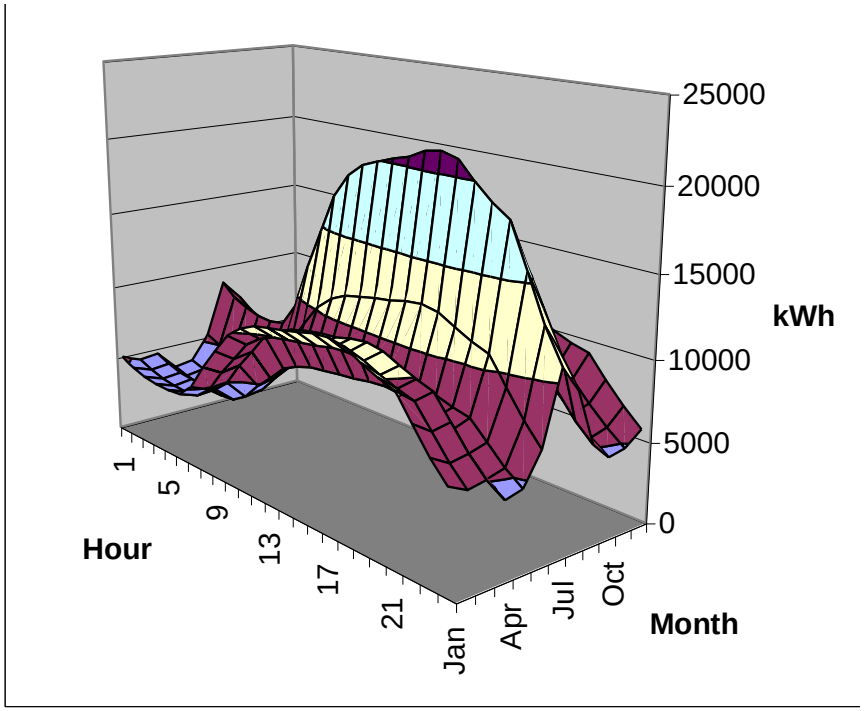
ELECTRIC POWER RESEARCH INSTITUTE

# Examples of Analysis Performed by DSS

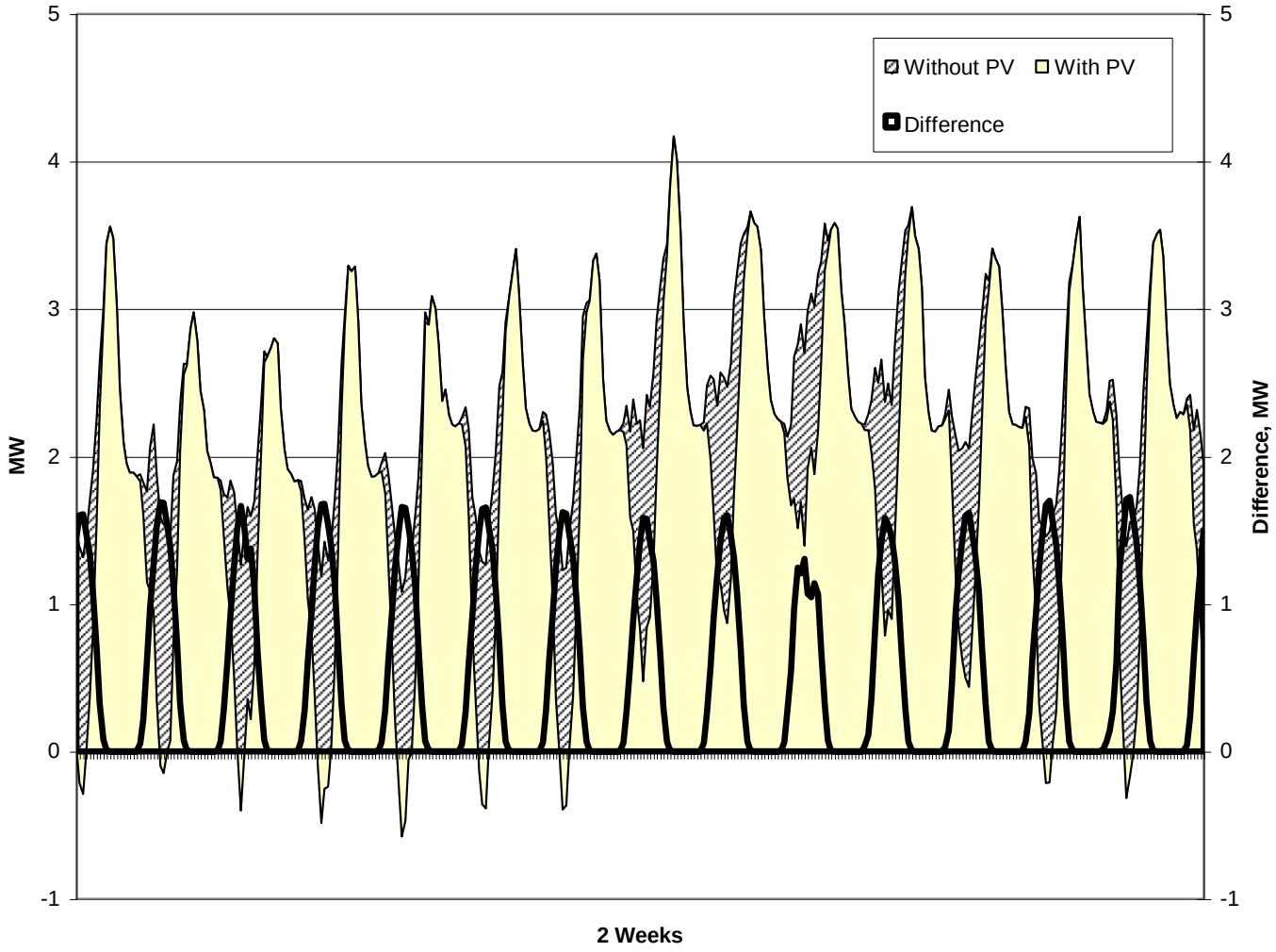
# Annual Losses

Peak load losses are not necessarily indicative of annual losses

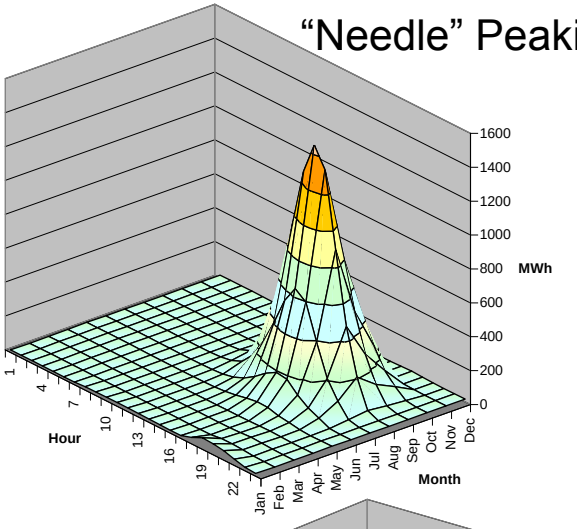


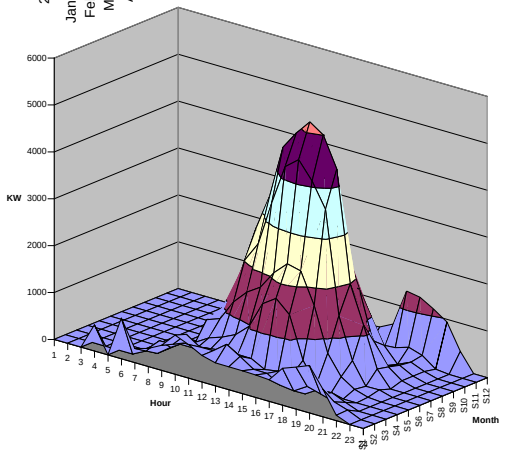



# Solar PV Simulation

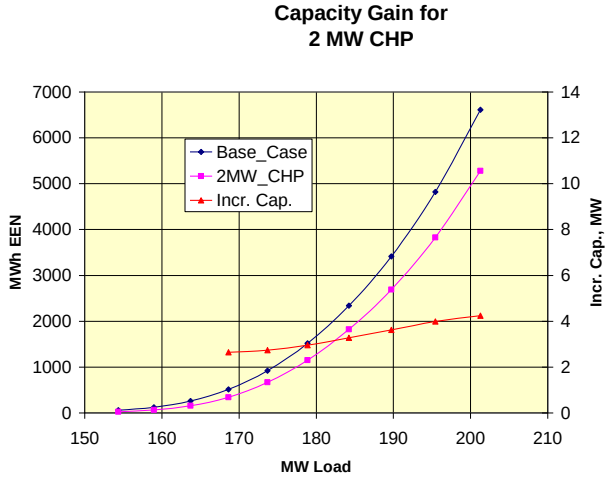


# Using DSS to Determine Incremental Capacity of DG




Broad Summer Peaking System



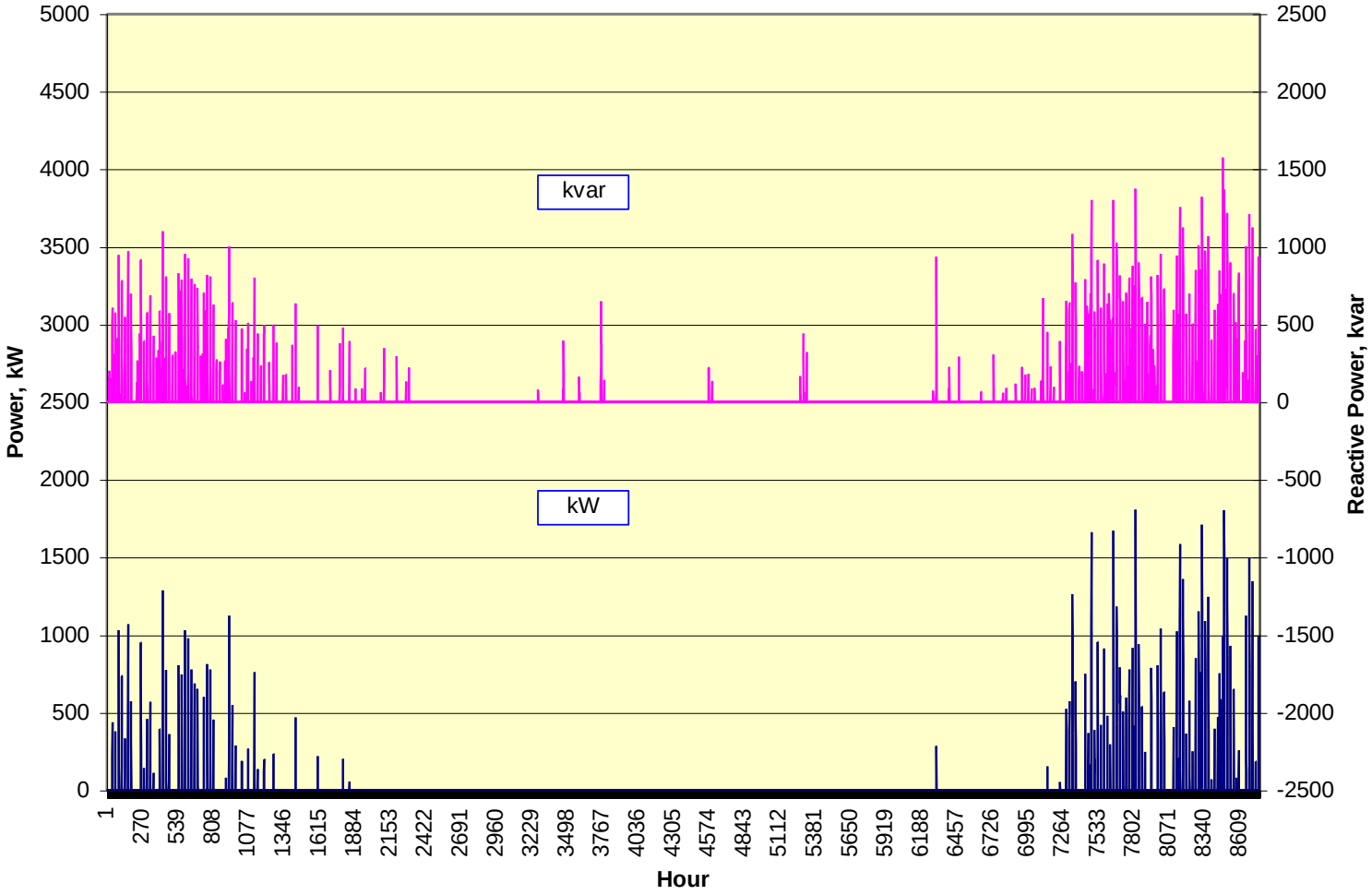

"How much more power can be served at the same risk of unserved energy?"



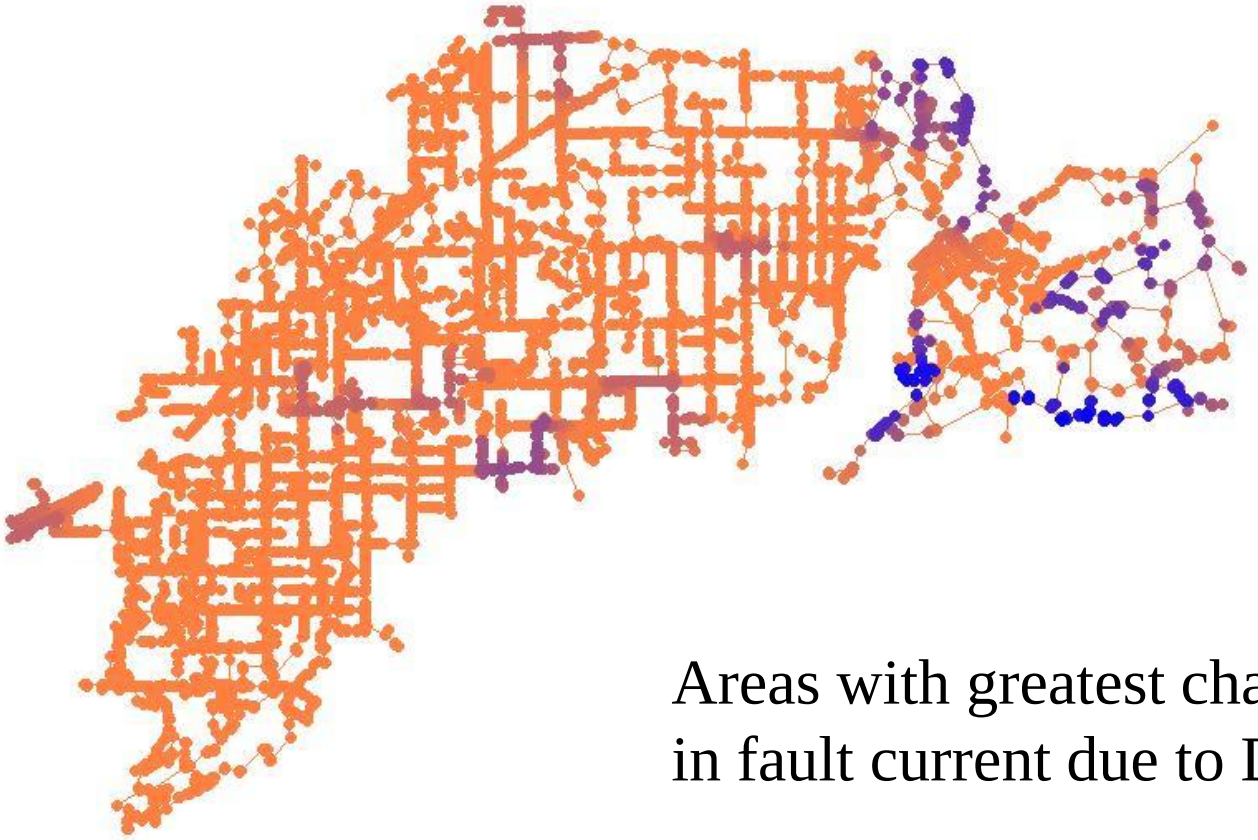


EPRI ELECTRIC POWER RESEARCH INSTITUTE

# DG Dispatch

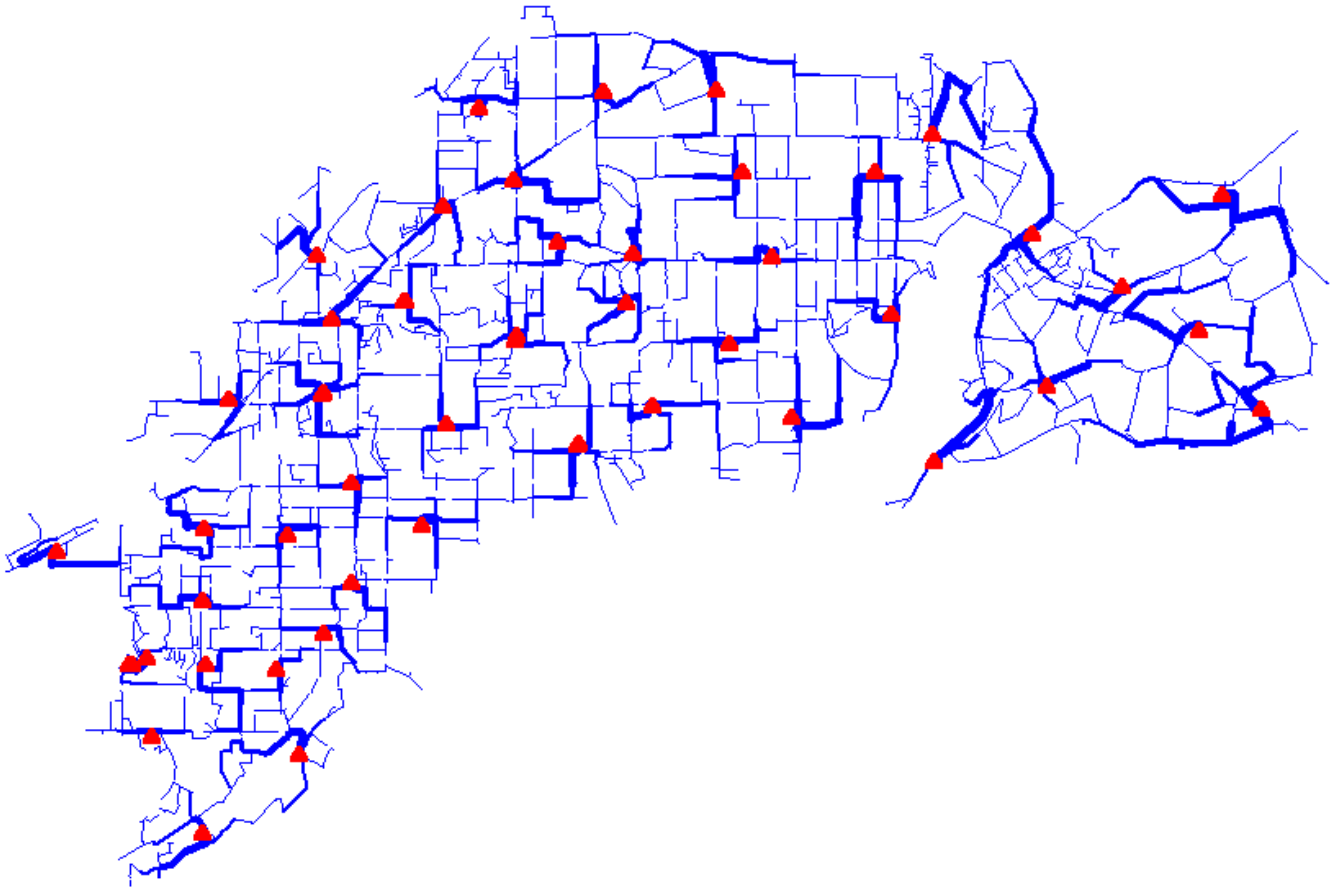


# DG Impact Visualization

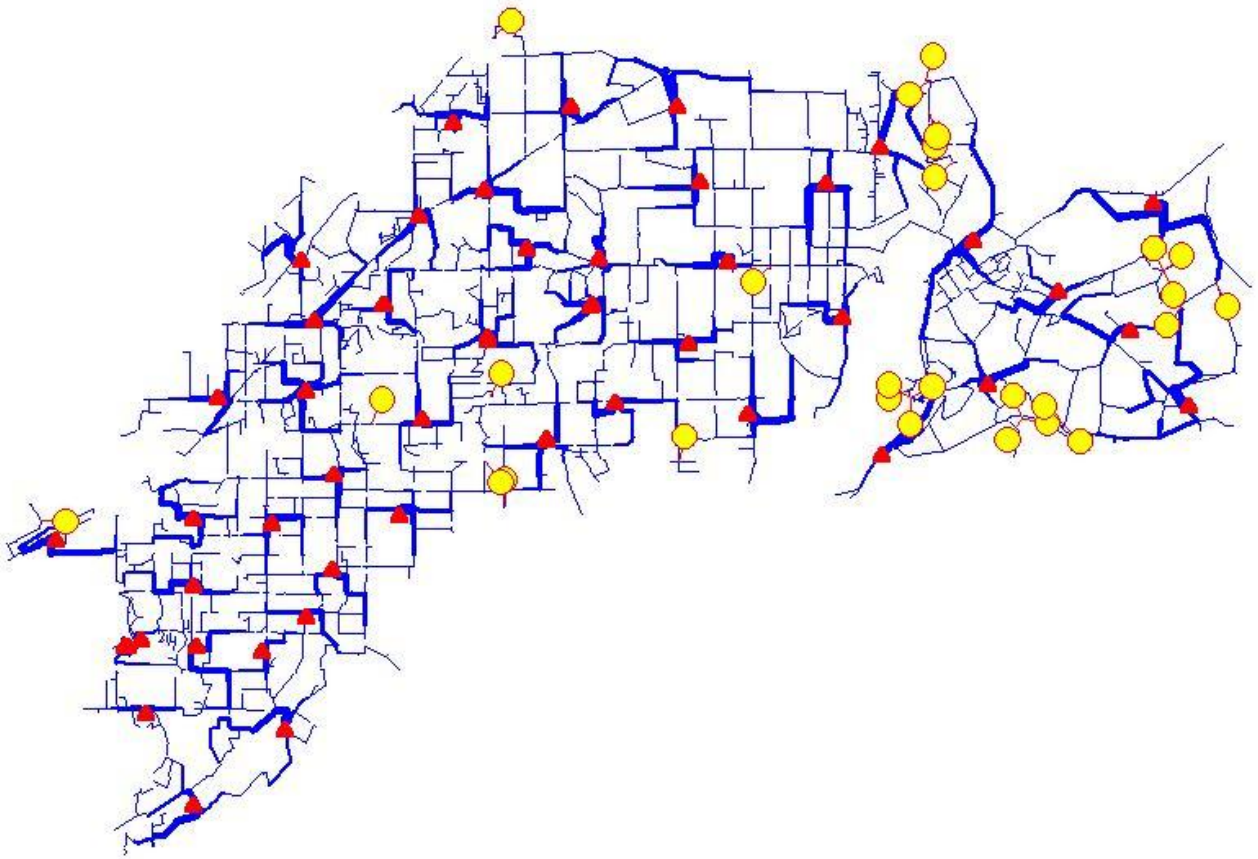


Areas with greatest change in fault current due to DG

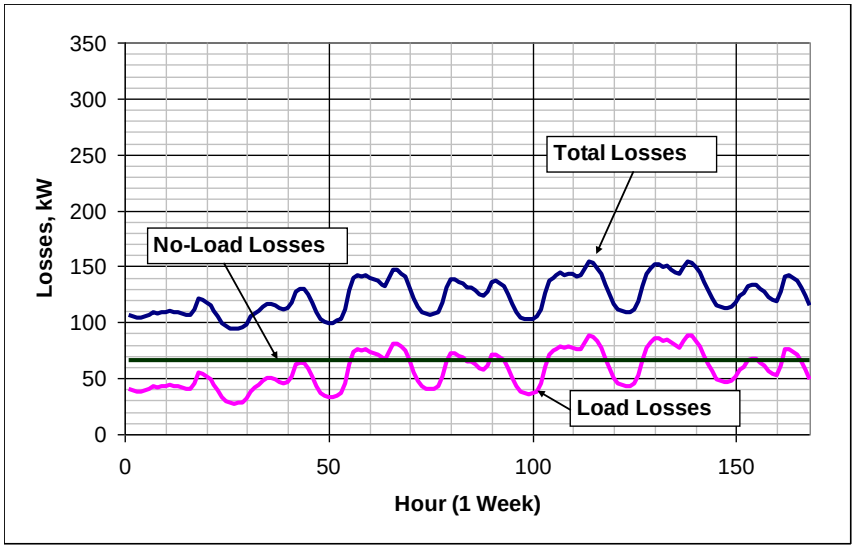
# Power Flow Visualization



# Optimal DG Siting

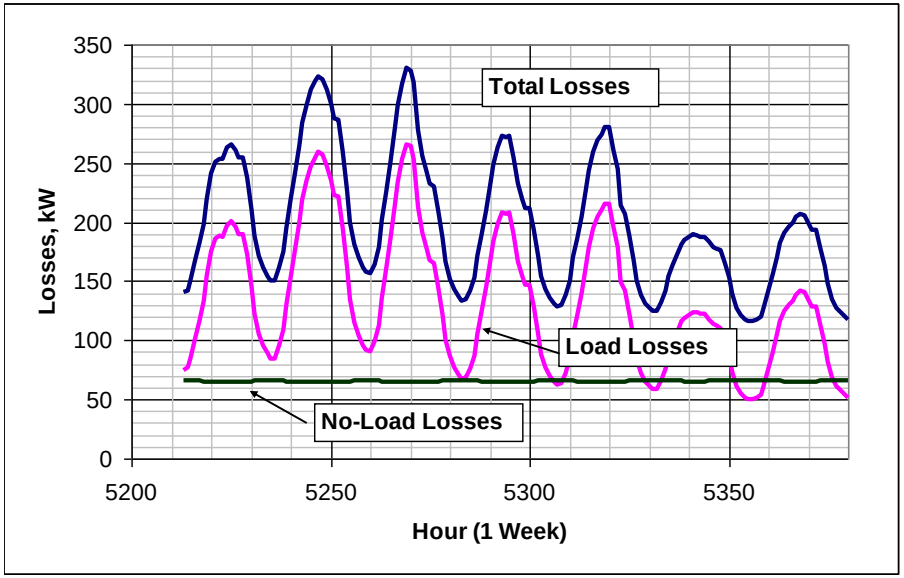


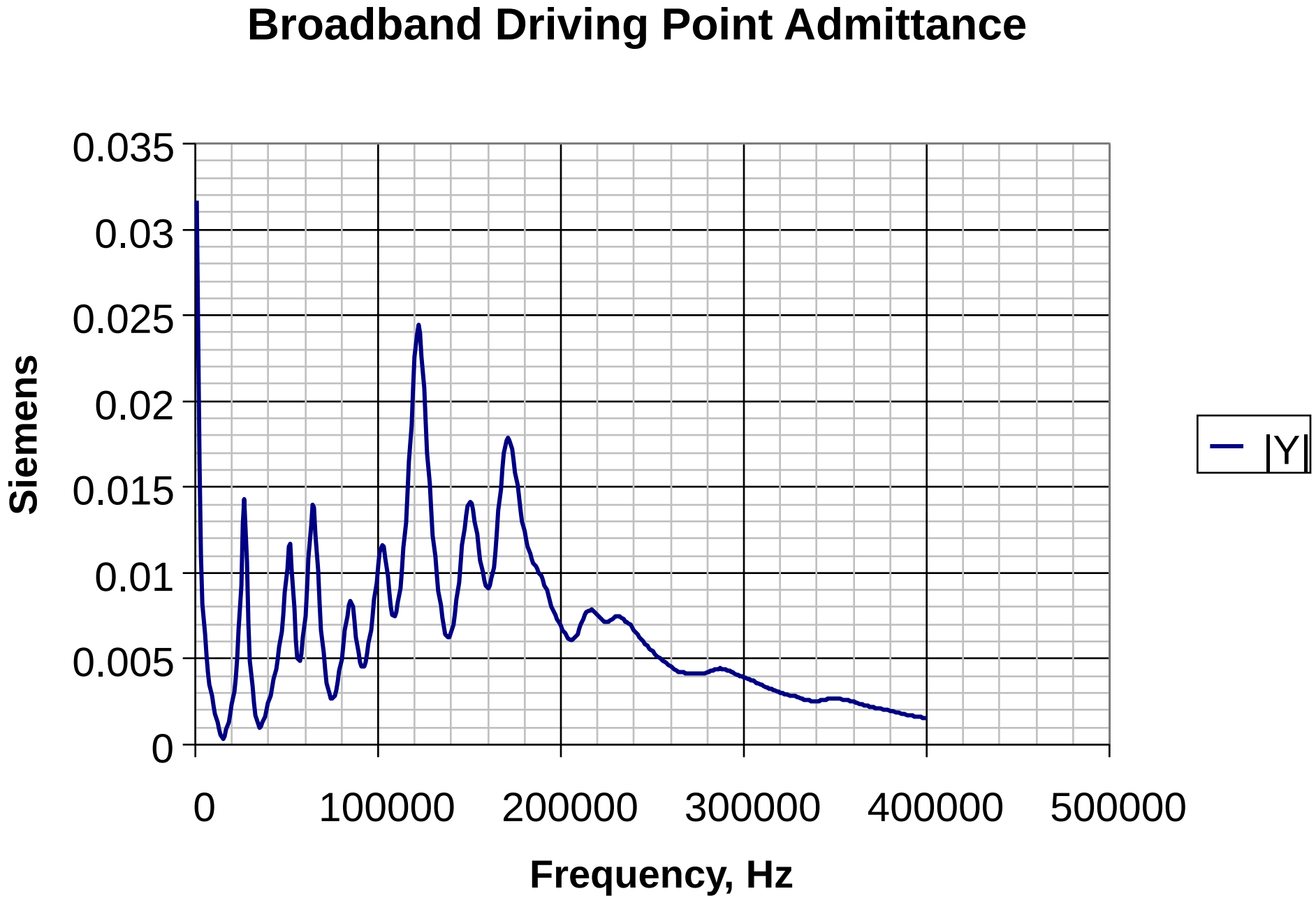
# Power Distribution Efficiency



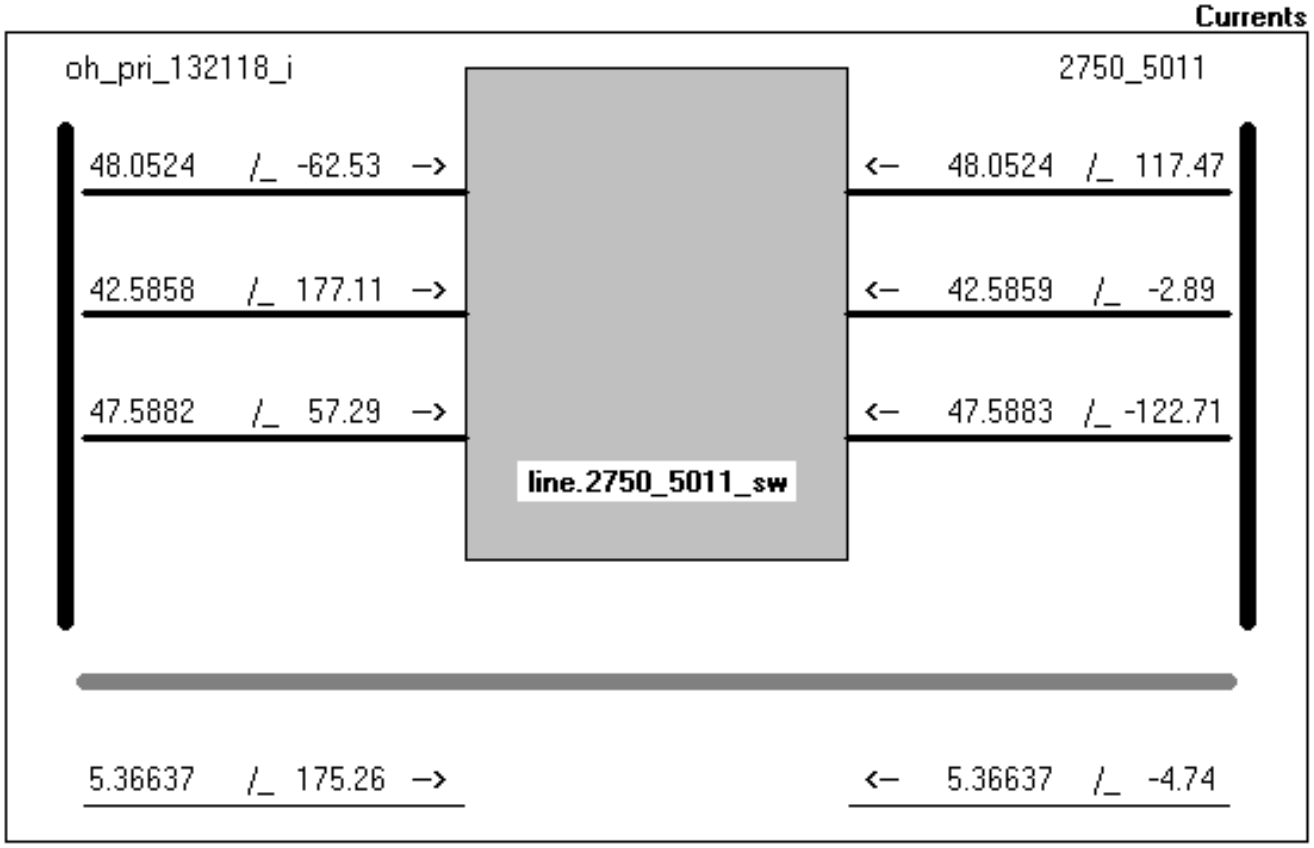

Light Load Week

Peak Load Week

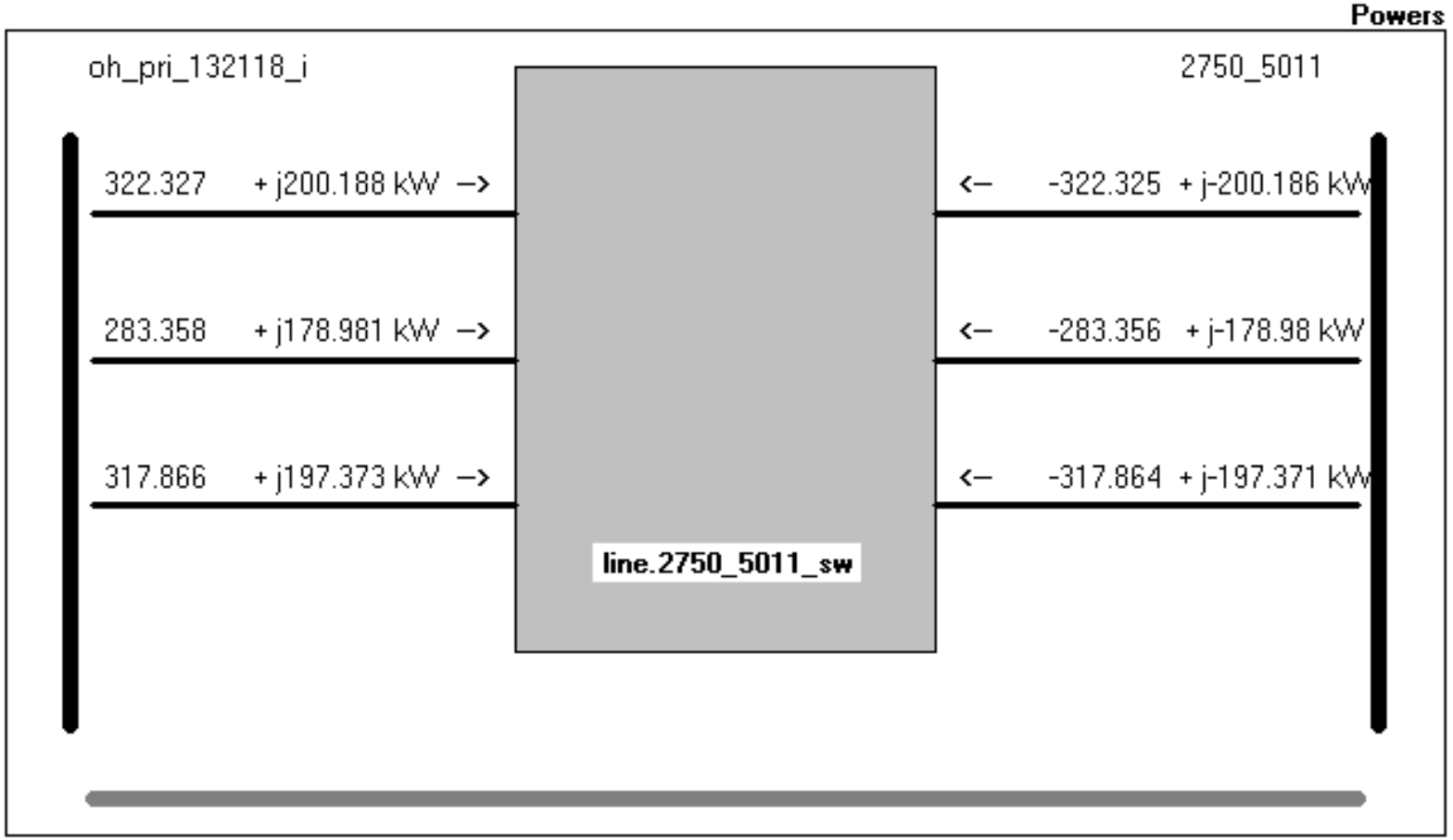



Broadband Driving Point Admittance
0.035
0.03
0.025
0.02
0.015
0.01
0.005
0
Siemens
0
100000
200000
300000
400000
500000
Frequency, Hz
|Y|

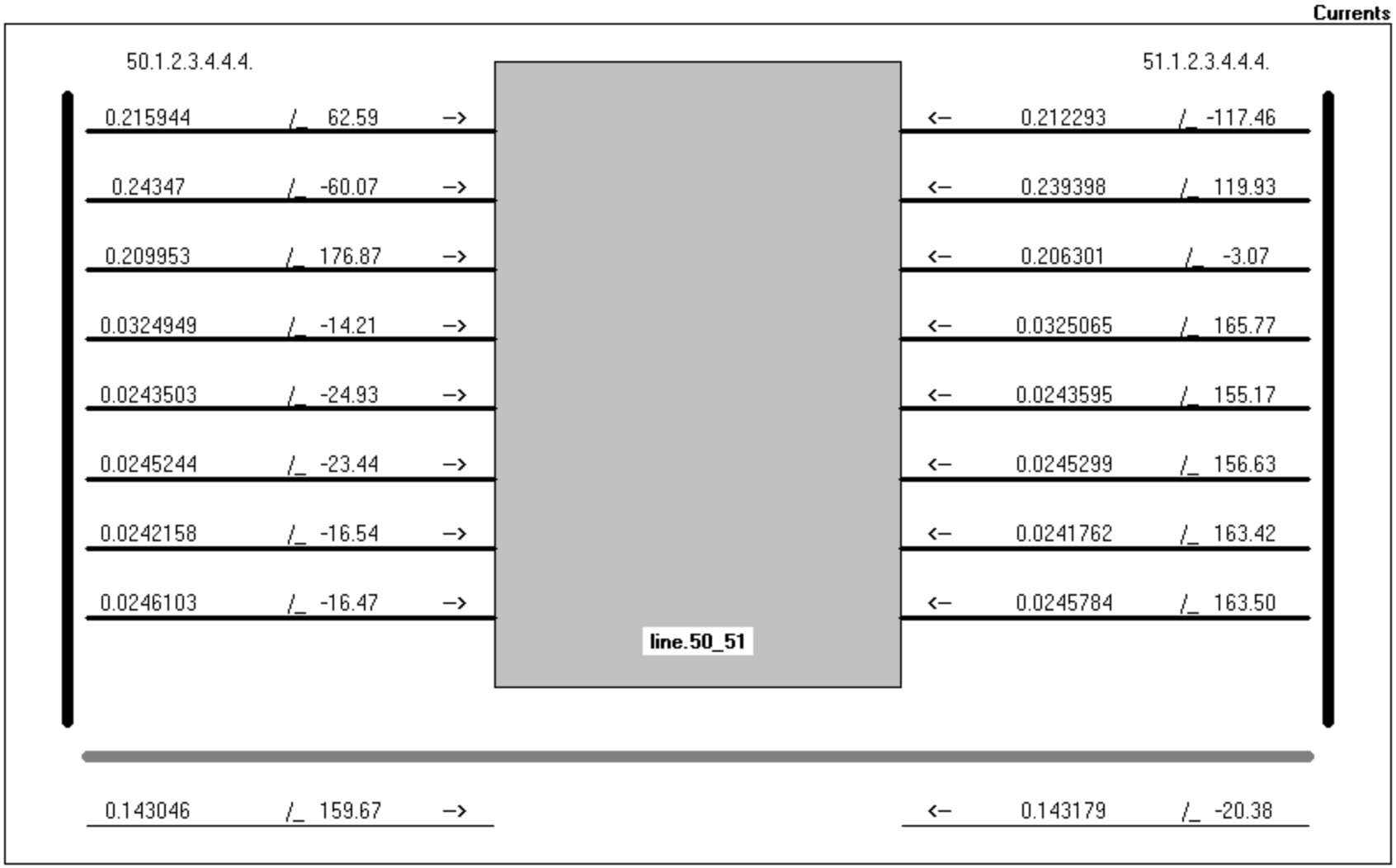
# Current



# Power

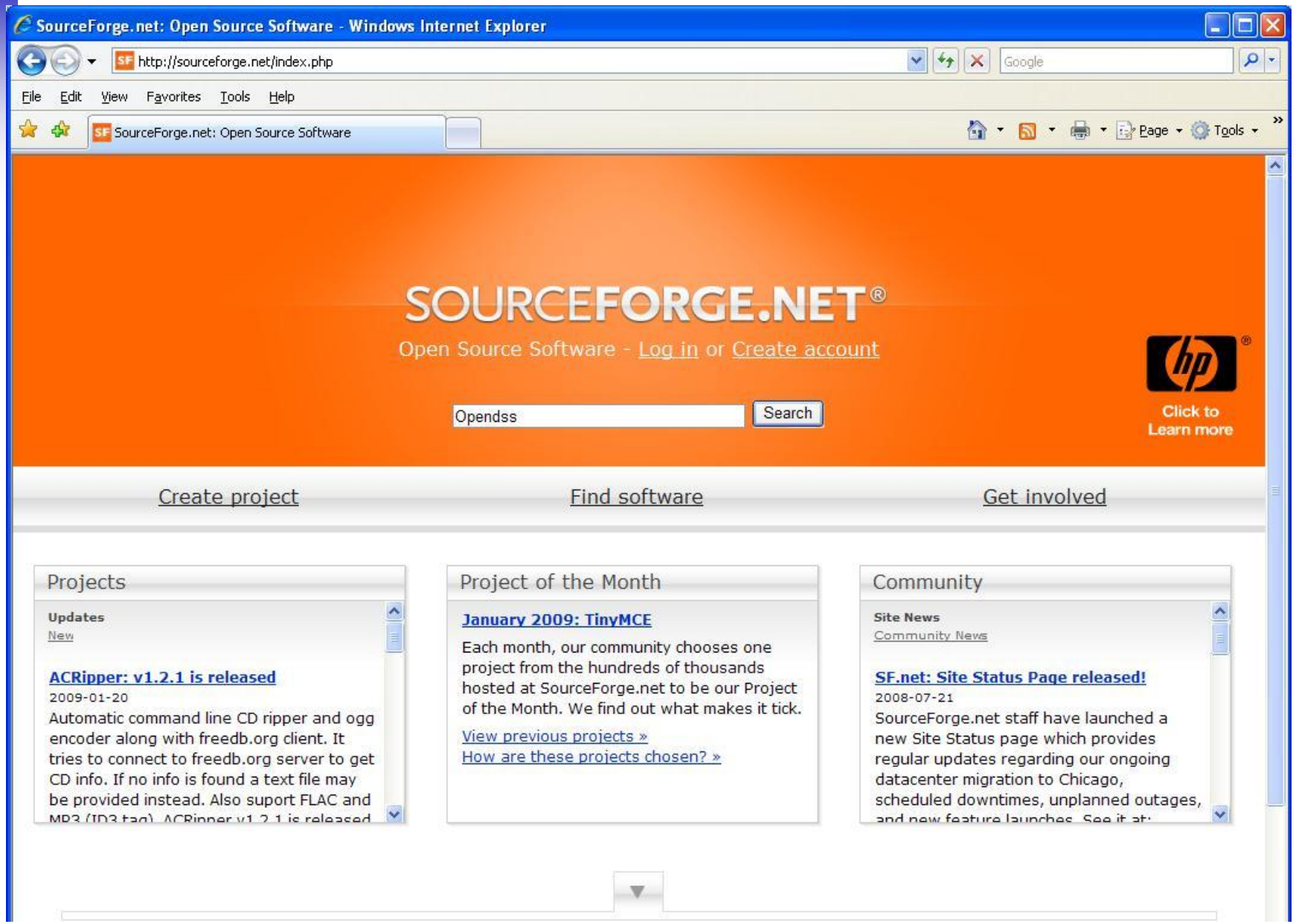


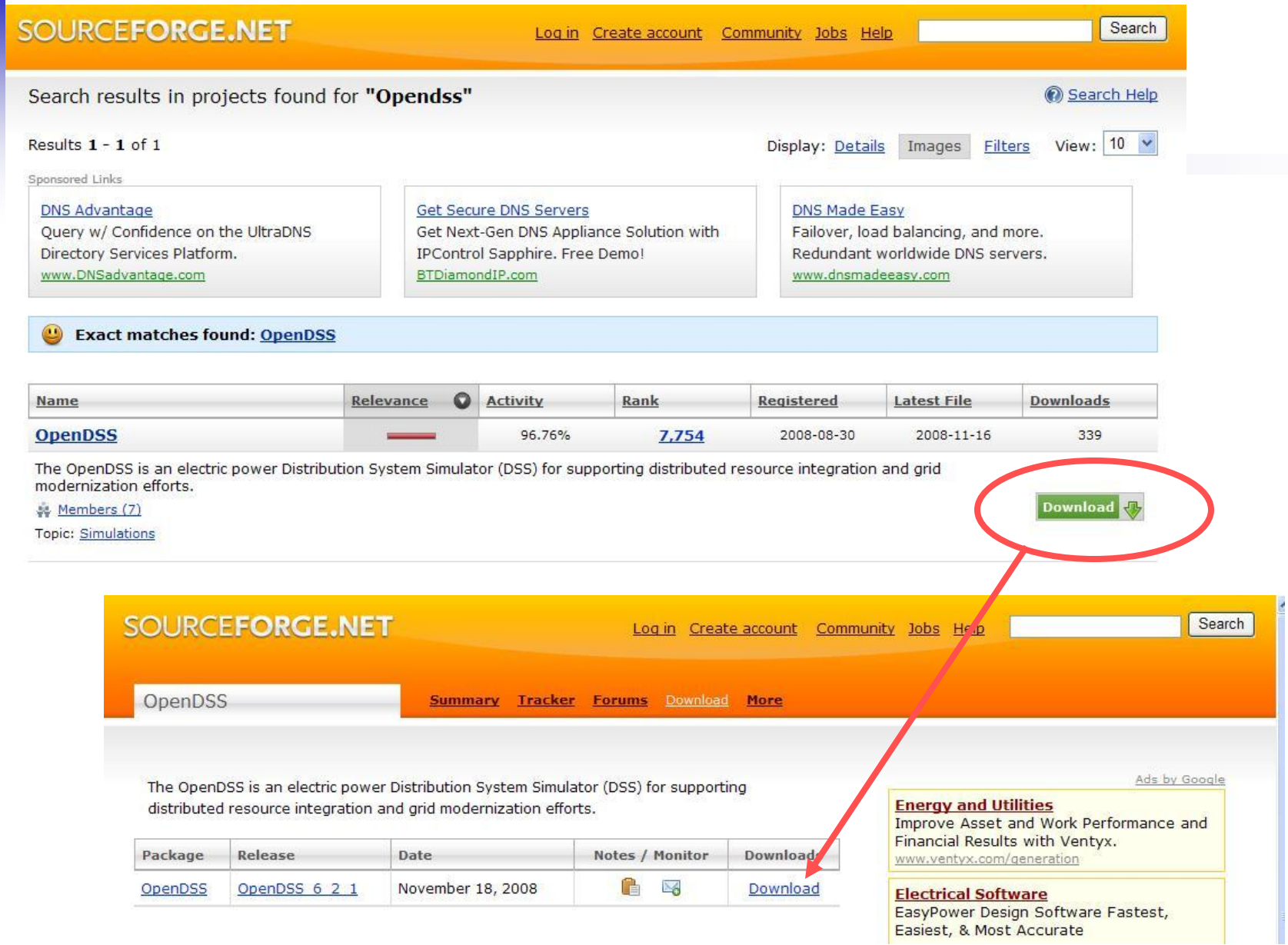
# A Bit More Complicated …



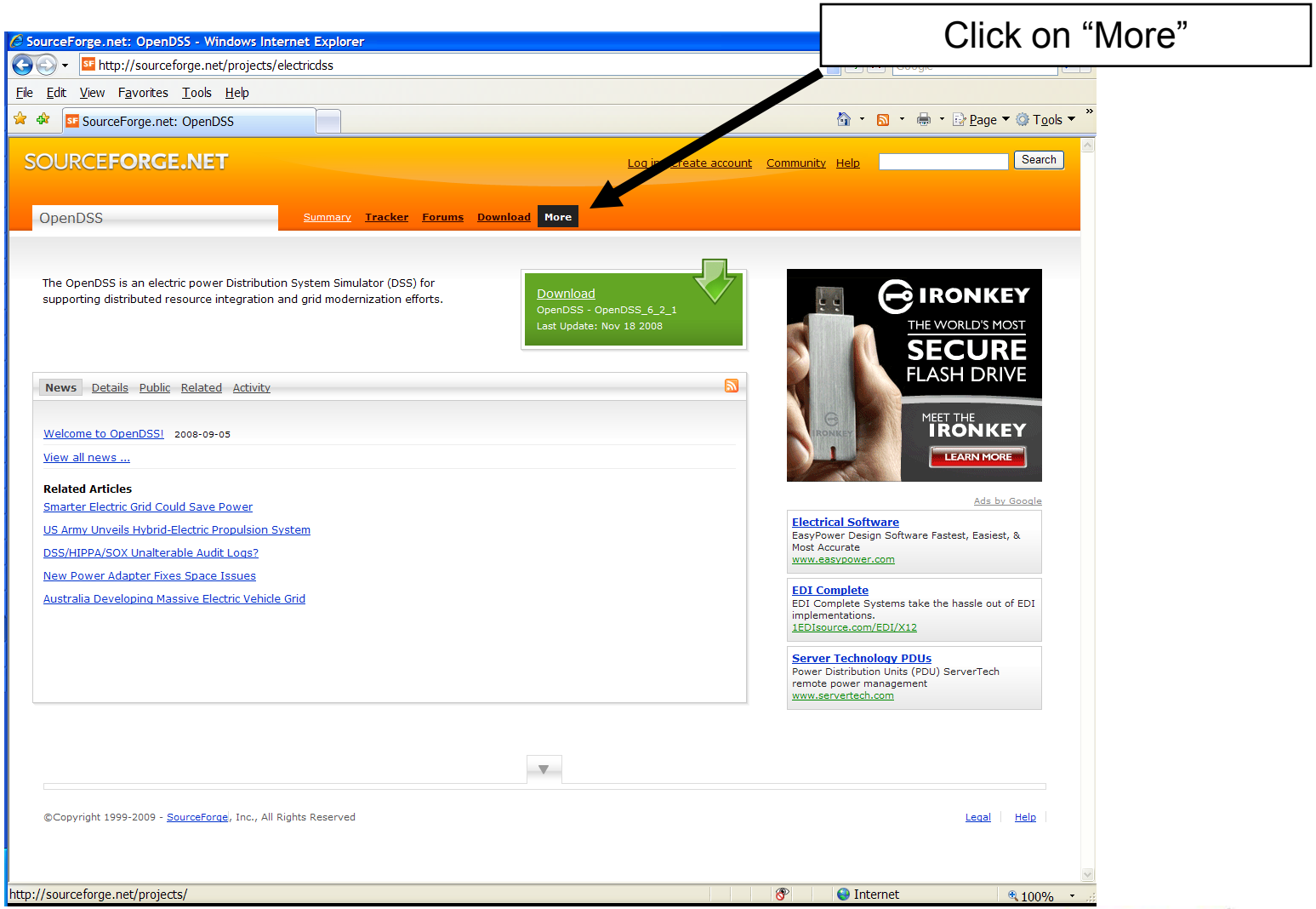
EPRI | ELECTRIC POWER RESEARCH INSTITUTE

# Installation

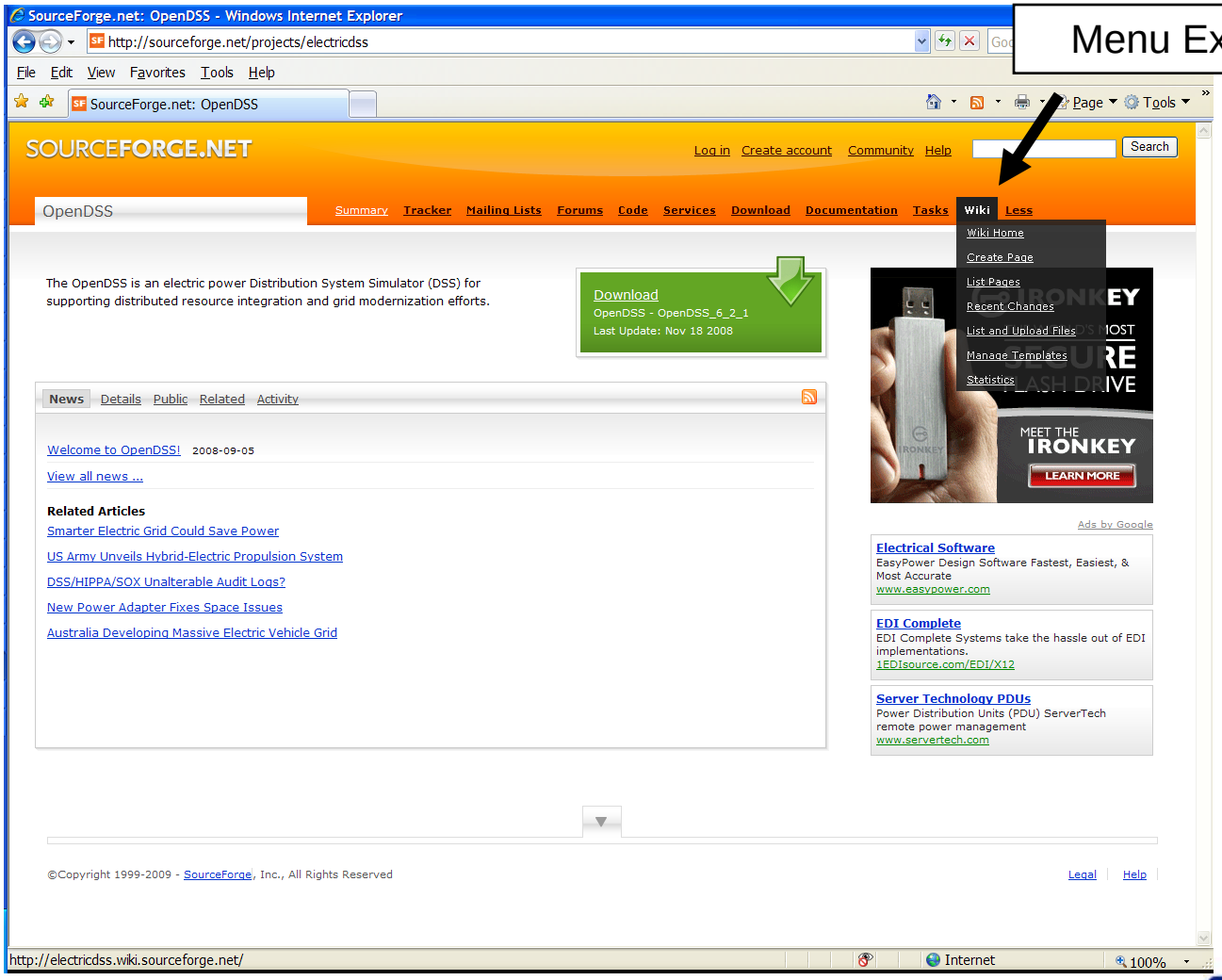




# Finding the Wiki …



# Finding the Wiki, cont'd



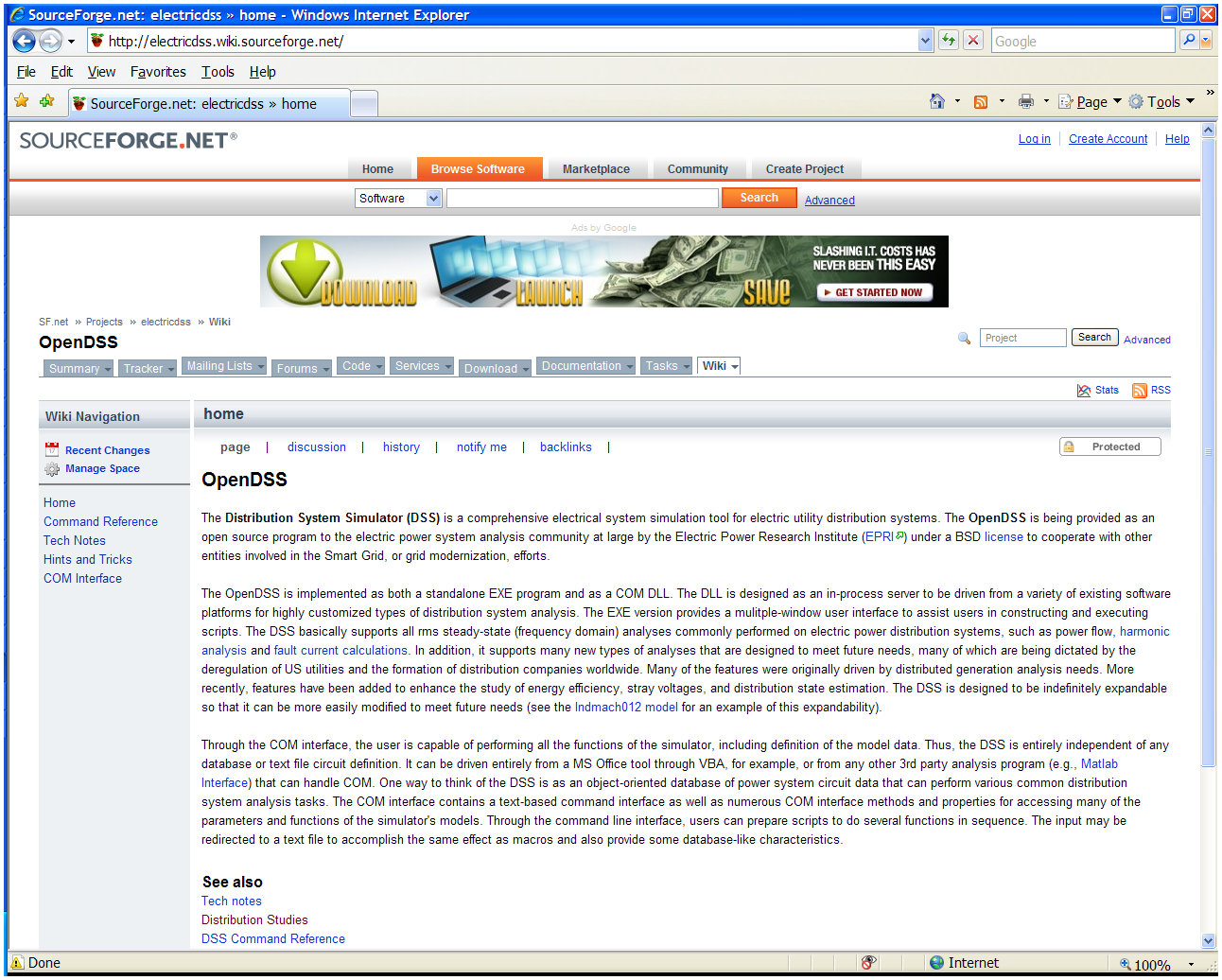

Menu Expands; Select Wiki





EPRI ELECTRIC POWER RESEARCH INSTITUTE

# Wiki Home Page (Latest documentation)








EPRI ELECTRIC POWER RESEARCH INSTITUTE

# Program Files

- OpenDSS.EXE — Standalone EXE
- OpenDSSEngine.DLL — In-process COM server
- KLUSolve.DLL — Sparse matrix solver
- DSSgraph.DLL — DSS graphics output

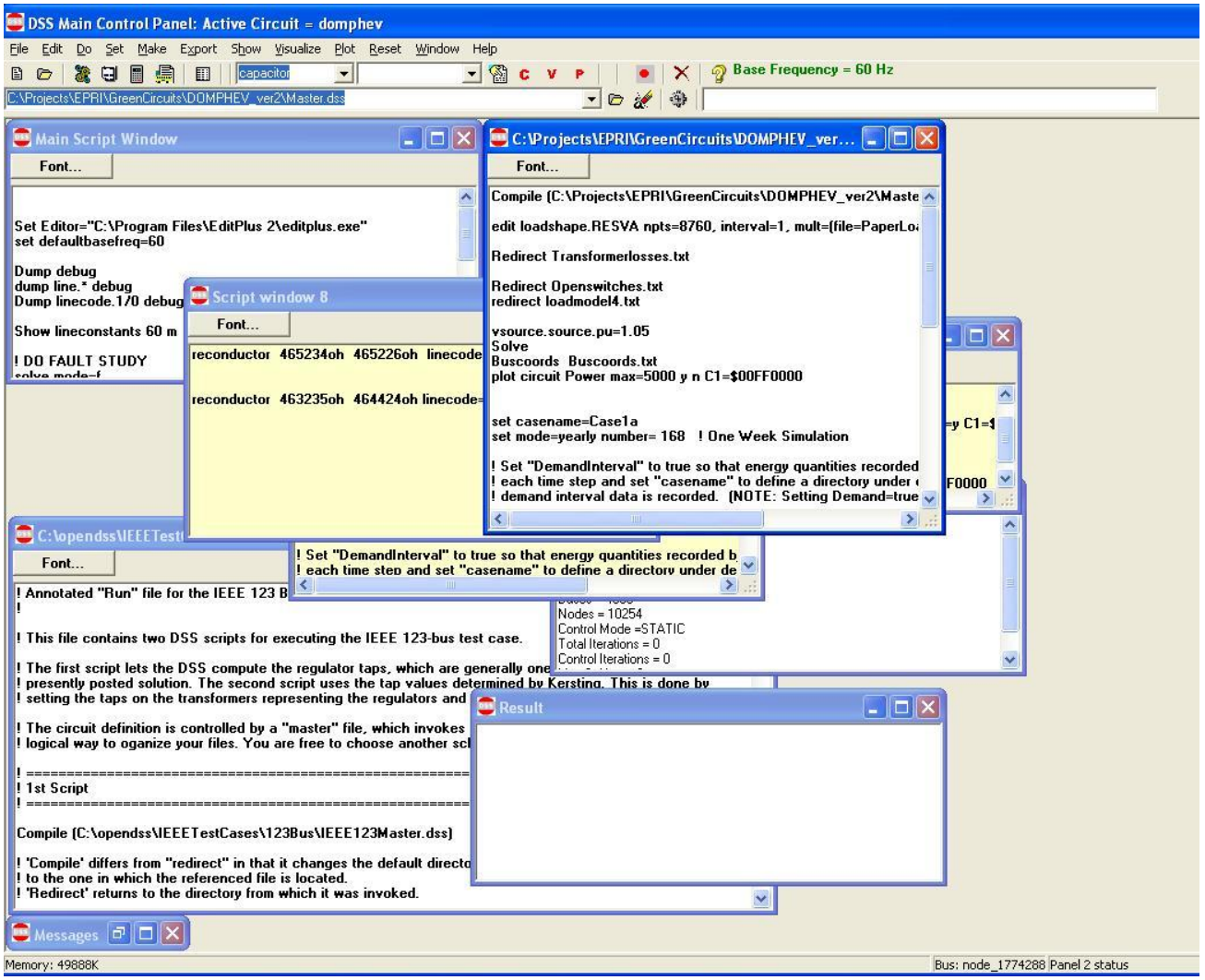
- Copy these files to the directory (folder) of your choice
  - Typically **`c:\OpenDSS`** *or* **`c:\Program Files\OpenDSS`**

- If you intend to drive OpenDSS from another program, you will need to register the COM server

# Registering the COM Server

- In DOS window, change to folder where you installed it and type:
  - **`Regsvr32 OpenDSSEngine.DLL`**
- The Server shows up as "**OpenDSSEngine.DSS**" in the Windows Registry
- For Example, to include in Matlab:
  - **`DSSobj = actxserver('OpenDSSEngine.DSS');`**
- In VBA:
  - **`Public DSSobj As OpenDSSEngine.DSS`**
    **`Set DSSobj = New OpenDSSEngine.DSS`**

# OpenDSS Standalone EXE User Interface

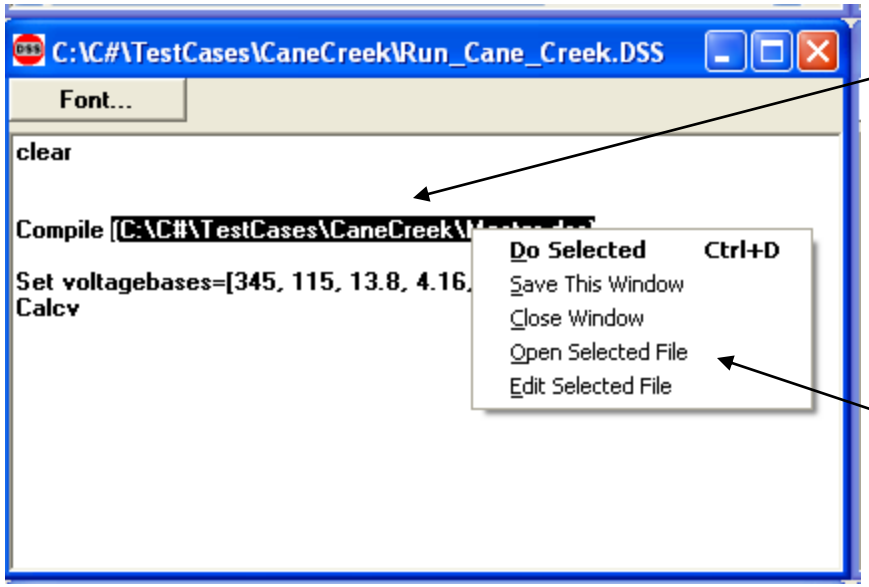







EPRI ELECTRIC POWER RESEARCH INSTITUTE

# Executing Scripts in the EXE



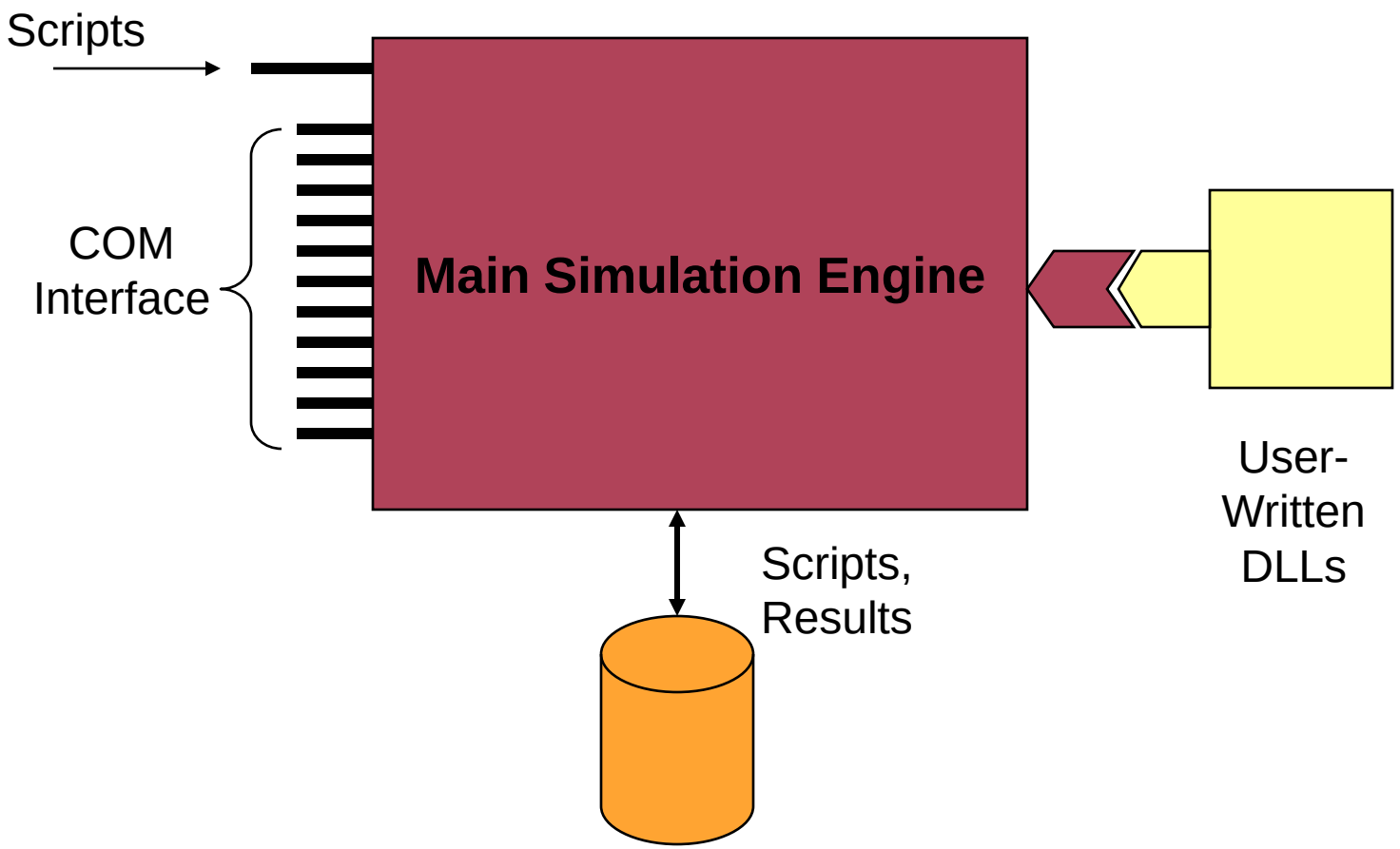

Select all or part of a line

Right-Click to get this pop-up menu

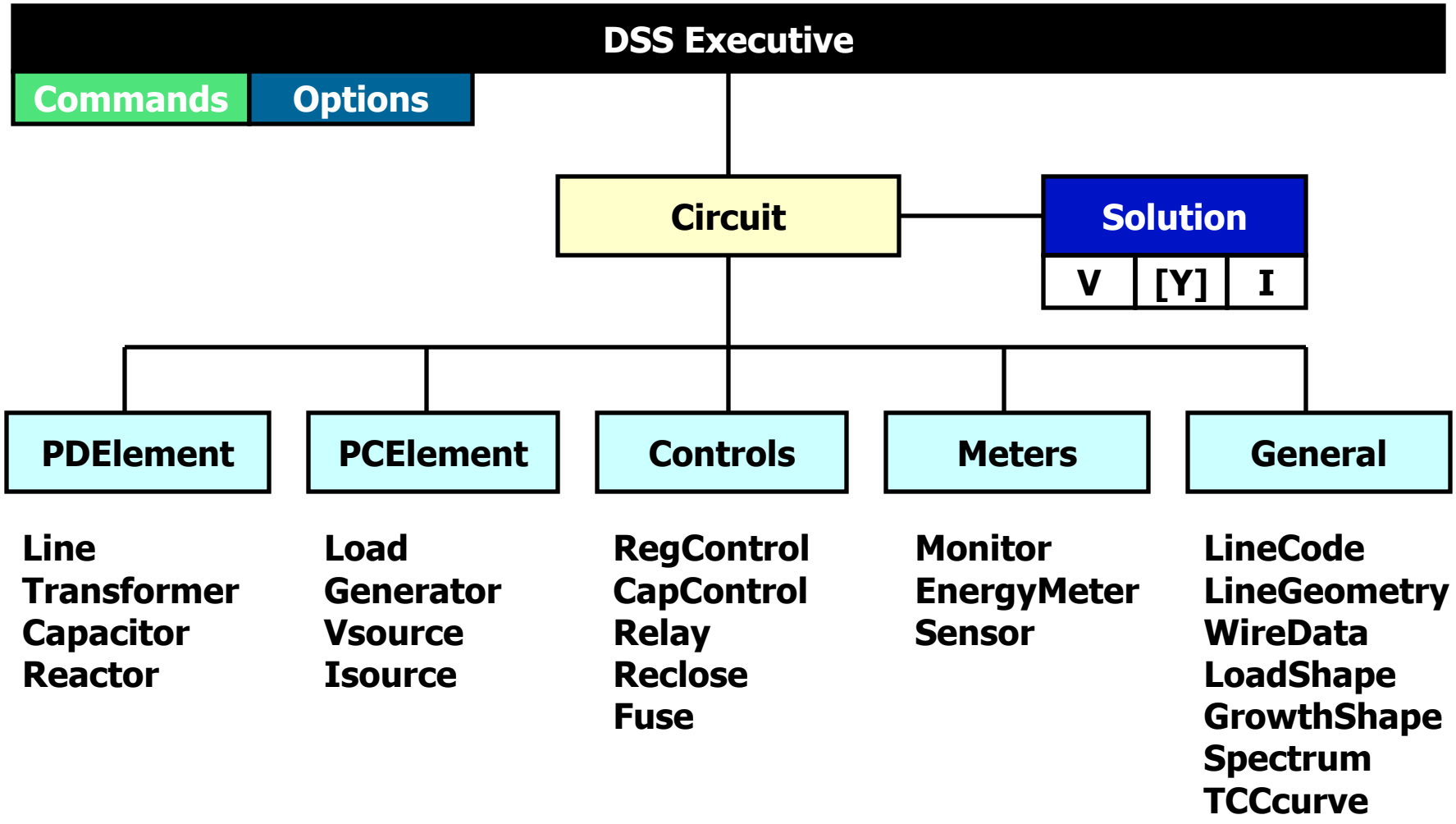
DSS executes selected line or opens selected file name

Any script window may be used at any time.

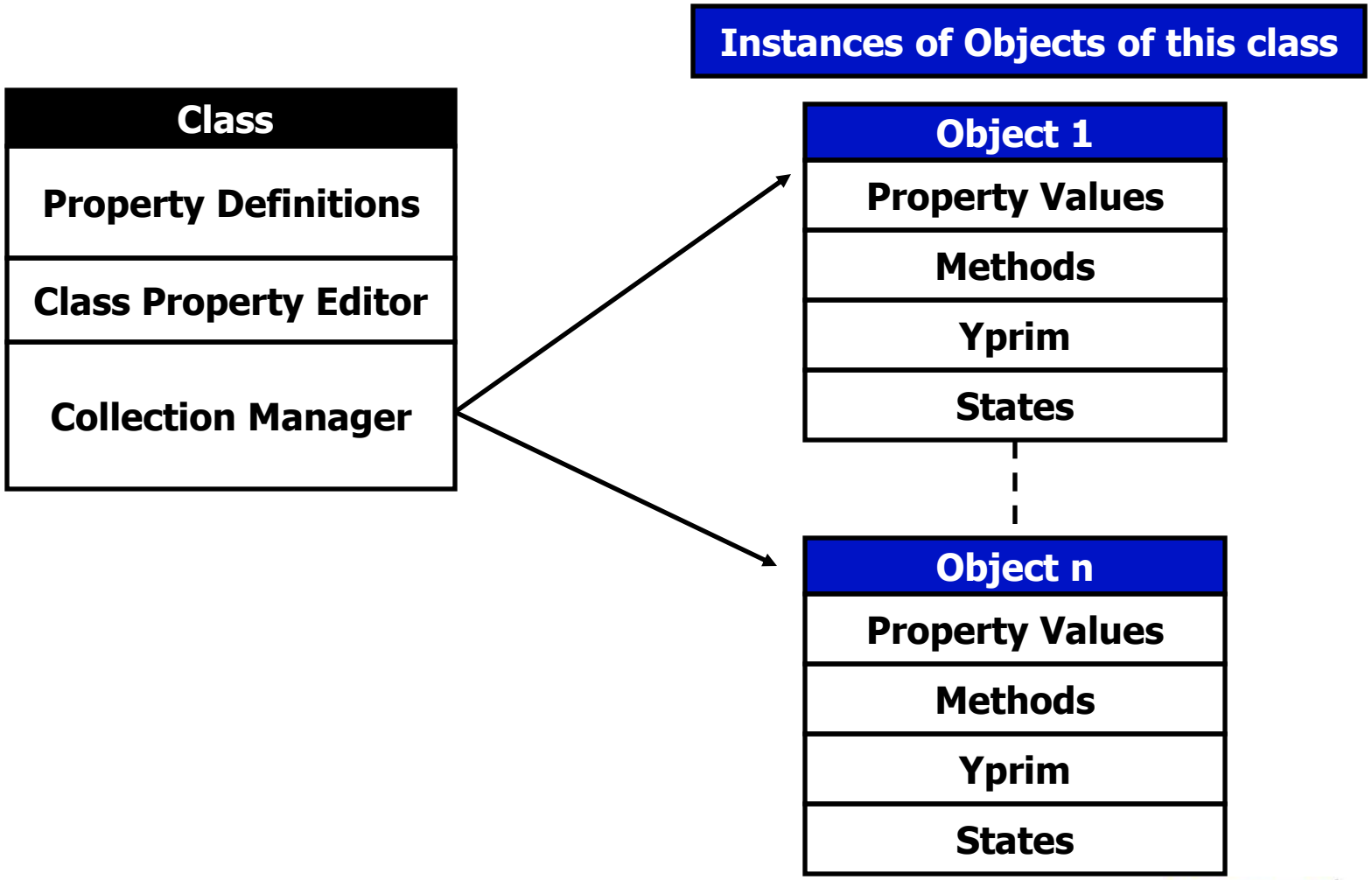
# DSS Structure



# DSS Object Structure



# DSS Class Structure



# DSS Classes

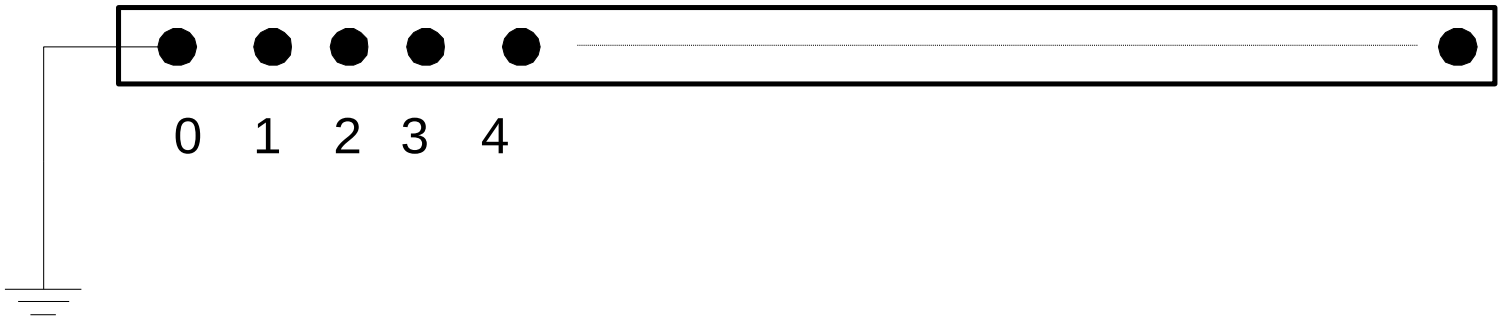
- Power Delivery (PD) Elements
  - **Line**
  - **Transformer**
  - **Reactor**
  - **Capacitor**
- Power Conversion (PC) Elements
  - **Load**
  - **Generator**
  - **Vsource**
  - **Isource**
- Control Elements
  - **RegControl**
  - **CapControl**
  - **Recloser**
  - **Relay**
  - **Fuse**

- Metering Elements
  - Monitor
  - EnergyMeter
  - Sensor
- General
  - LineCode
  - LineGeometry
  - Loadshape
  - Growthshape
  - Wiredata
  - Spectrum
  - TCC Curves

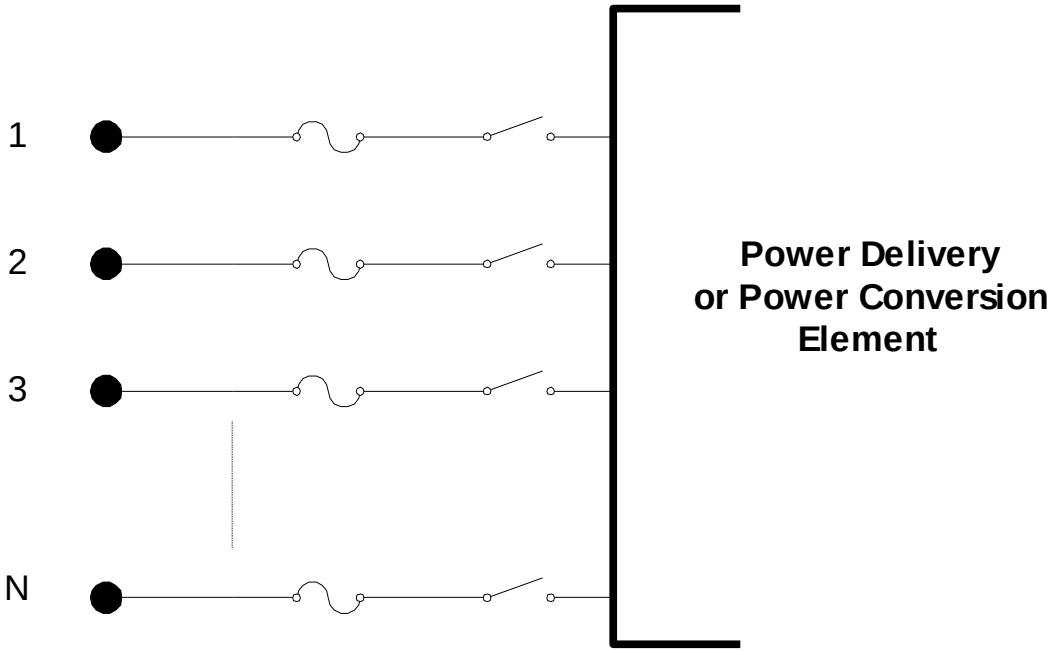
EPRI | ELECTRIC POWER RESEARCH INSTITUTE

# Circuit Principles

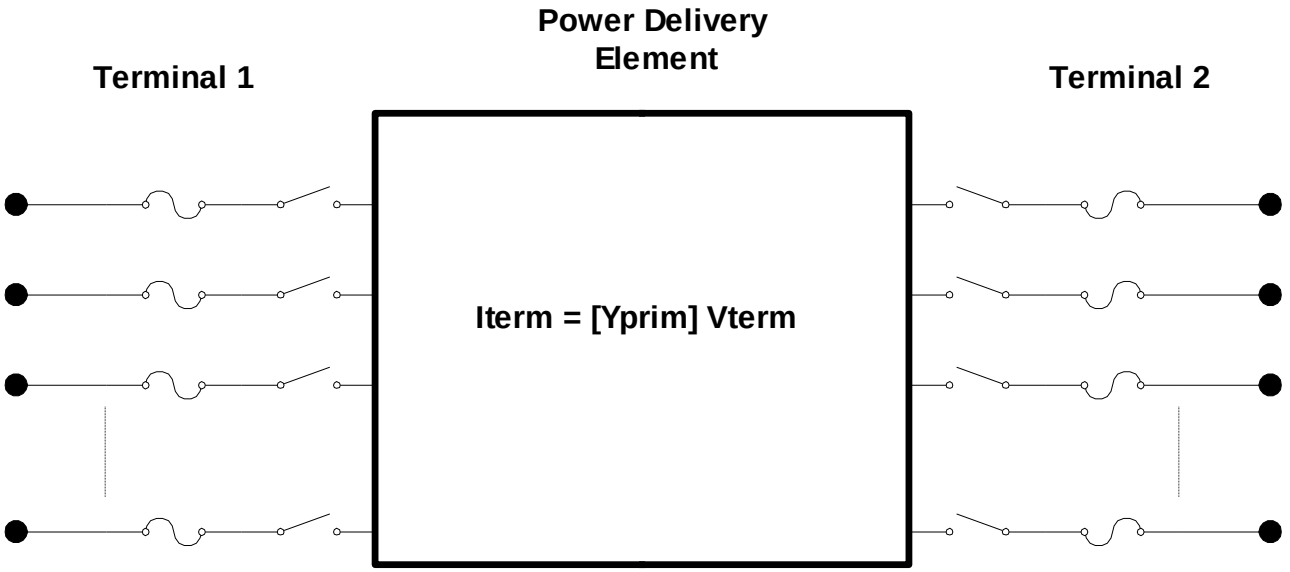
# DSS Bus Model



# DSS Terminal Definition

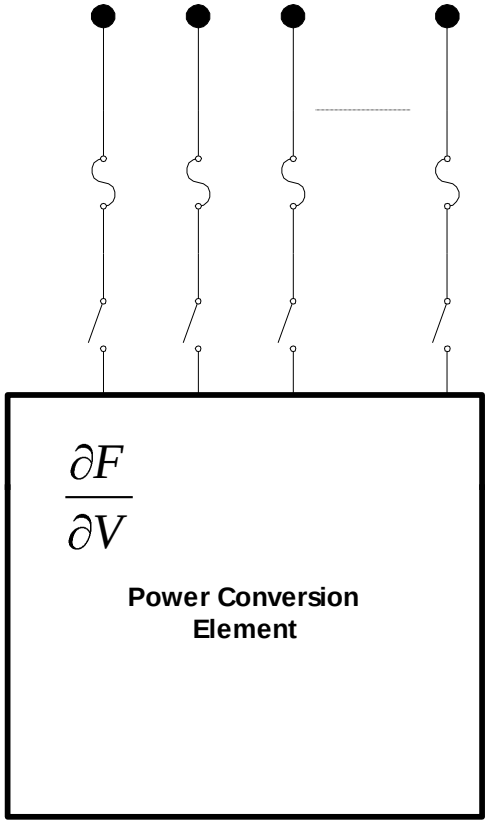


# Power Delivery Elements

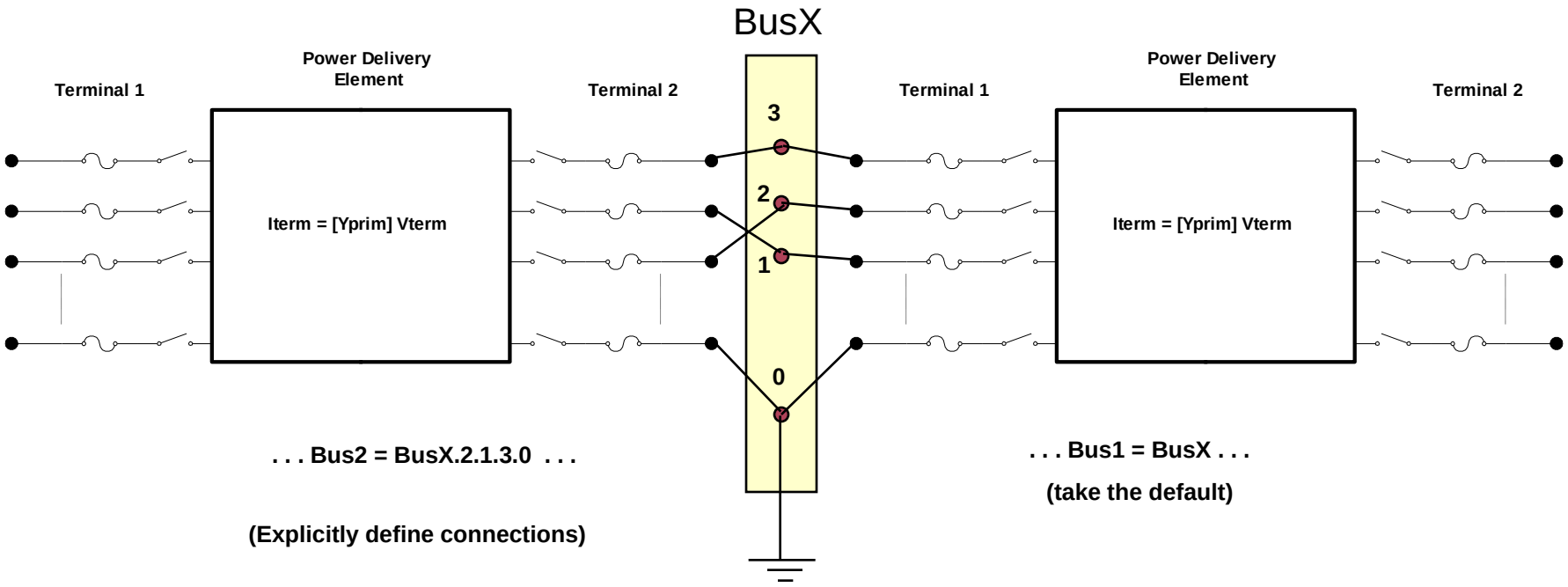


# Power Conversion Elements

$I_{Term}(t) = \mathbf{F}(V_{Term}, [State], t)$



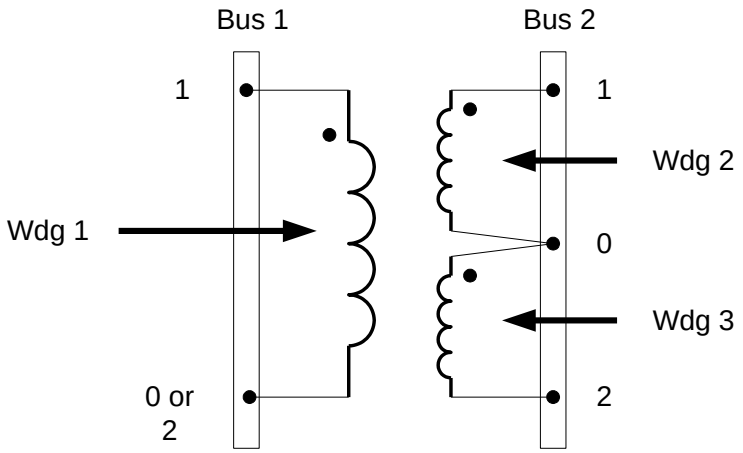
# Circuit Elements are Connected together at the Nodes of Buses




DSS Convention: A *Terminal* can be connected to only one *Bus*. You can have any number of *Nodes* at a bus.

# Connections for 1-Phase Residential Transformer

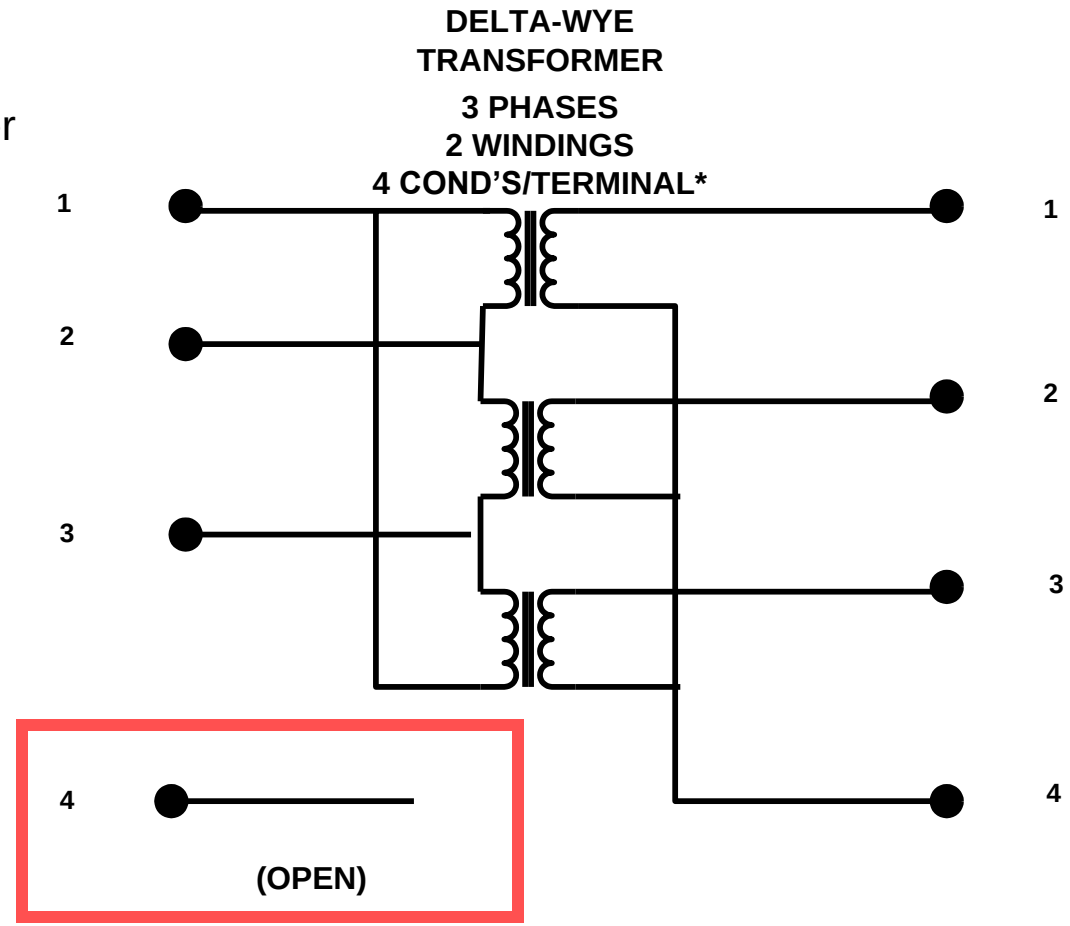
```
! Line-to-Neutral Connected 1-phase Center-tapped transformer
New  Transformer.Example1-ph  phases=1  Windings=3
~ Xhl=2.04  Xht=2.04  Xlt=1.36  %noloadloss=.2
~ Buses=[bus1.1 bus2.1.0  bus2.0.2]   !!! Note polarity
~ kVs=[7.2 .12  .12]     ! ratings of windings
~ kVAs=[25 25 25]
~ %Rs = [0.6  1.2  1.2]
~ conns=[wye wye wye]    ! default
```




Center-Tapped 1-Phase Transformer Model

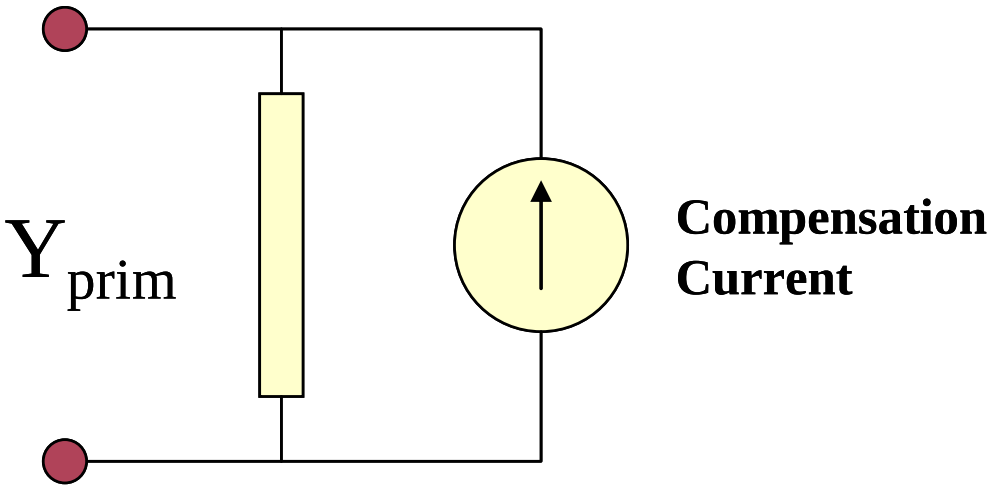
# All Terminals of a Circuit Element Have Same Number of Conductors

3-Phase Transformer



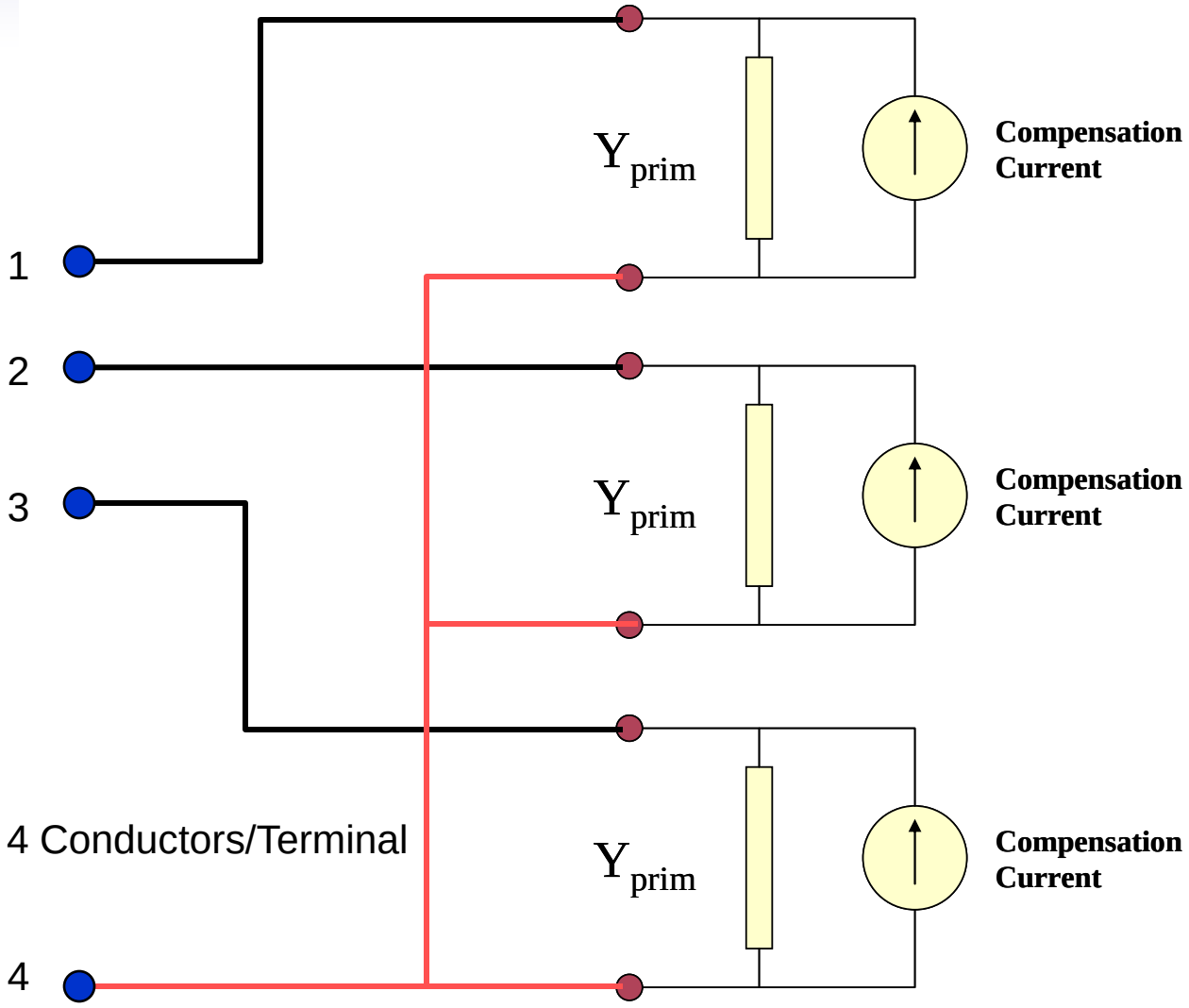

*** MUST HAVE THE SAME NUMBER OF CONDUCTORS FOR EACH TERMINAL**

# Load (a PC Element)

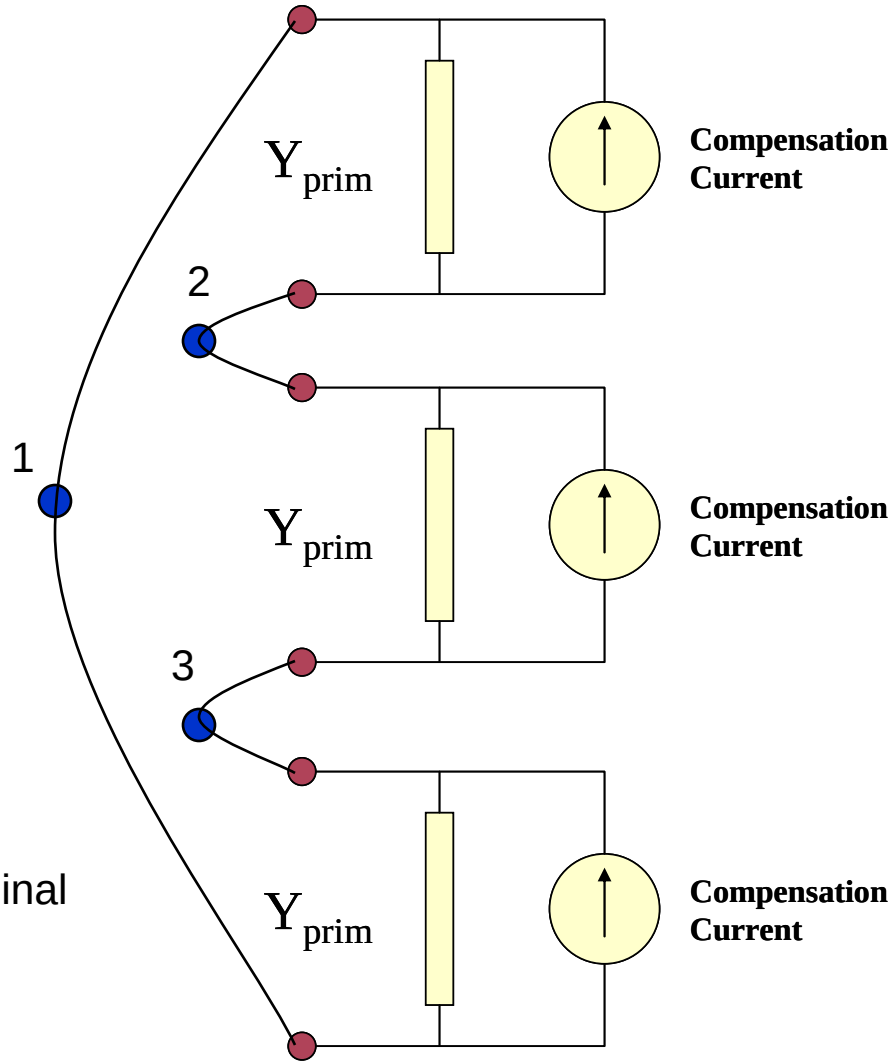



(One-Line Diagram)

# Load - 3-phase Y connected

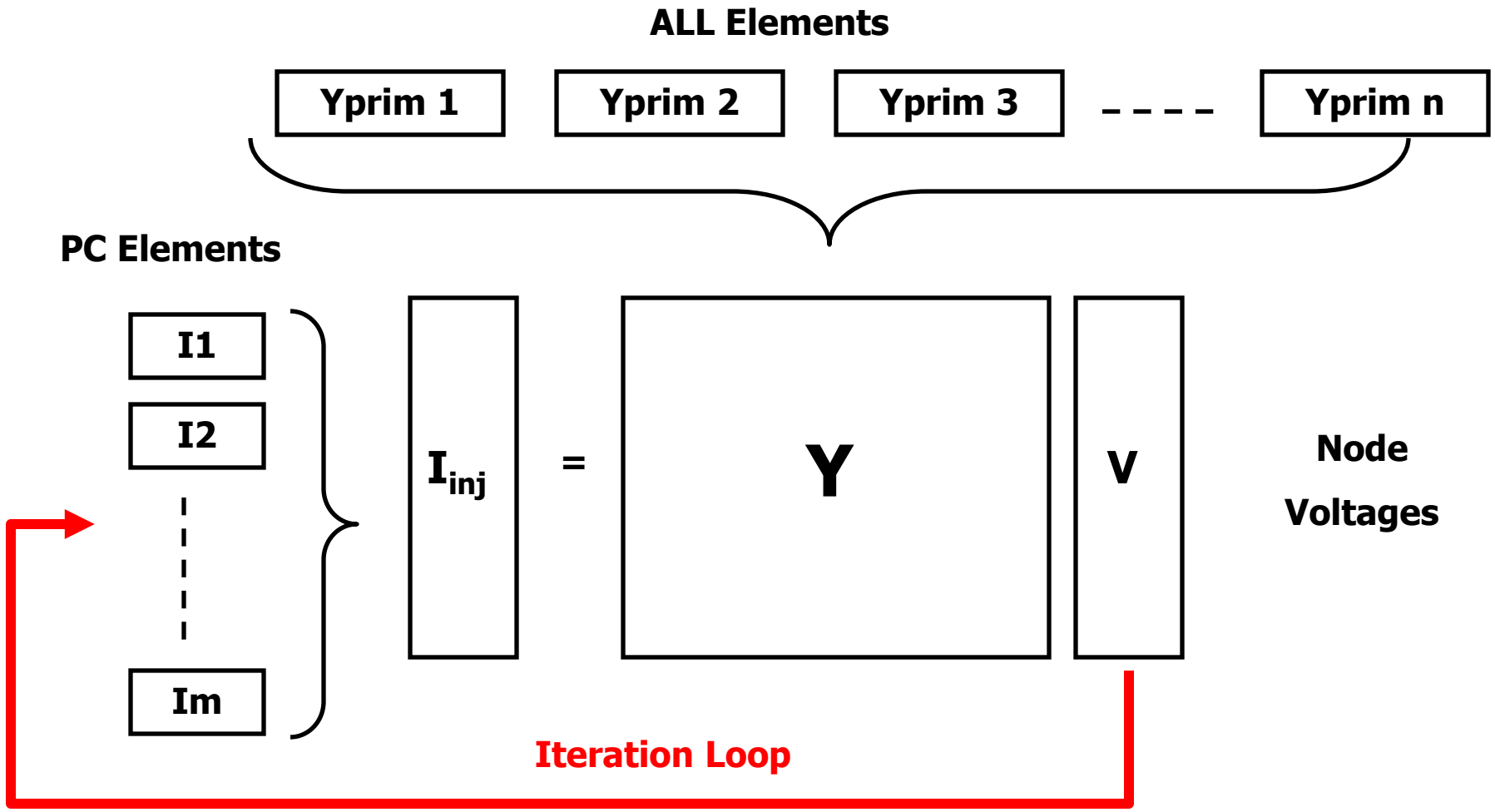


# Load - 3-phase Delta connected




3 Conductors/Terminal

# Putting it All Together

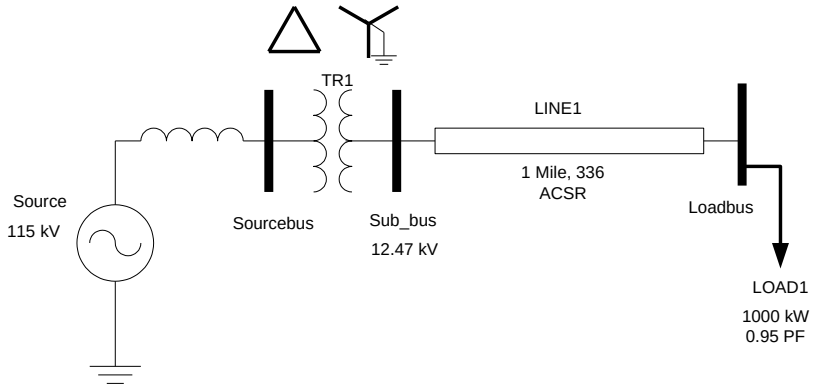


# Solution Speed

- Distribution systems generally converge quite rapidly with this method.
- The OpenDSS program seems to be on par with the faster commercial programs – or faster
- It is set up to run annual simulations easily
  - Our recommendation:
    - *Err on the side of running more power flow simulations*
    - That is, don't worry about the solution time until it proves to be a problem
    - That reveals more information about the problem.


# Scripting Basics

# A Basic Script




```
New Circuit.Simple     ! Creates voltage source  (Vsource.Source)
Edit Vsource.Source BasekV=115 pu=1.05  ISC3=3000  ISC1=2500  !Define source V and Z
New Transformer.TR1 Buses=[SourceBus, Sub_Bus] Conns=[Delta Wye] kVs= [115 12.47]
~ kVAs=[20000 20000] XHL=10
New Linecode.336ACSR R1=0.058 X1=.1206 R0=.1784 X0=.4047 C1=3.4 C0=1.6 Units=kft
New Line.LINE1 Bus1=Sub_Bus Bus2=LoadBus Linecode=336ACSR Length=1 Units=Mi
New Load.LOAD1 Bus1=LoadBus kV=12.47 kW=1000 PF=.95
Solve
Show Voltages
Show Currents
Show Powers kVA elements
```

# Command Syntax

- *Command parm1, parm2 parm3 parm 4 ….*

- Parameters may be positional or named (tagged).
- If named, an "**=**" sign is expected.
  - ***Name=value*** *(this is the named form)*
  - ***Value*** *(value alone in positional form)*
- *For example, the following two commands are equivalent:*

  - ***New Object="Line.First Line" Bus1=b1240 Bus2=32 LineCode=336ACSR, …***
  - ***New "Line.First Line", b1240 32 336ACSR, …***

Comma or white space

# Delimiters

- Array or string delimiter pairs: **[ ] , { },( ),“ “,‘ ‘**
- Matrix row delimiter: **|**
- Value delimiters: **,** (comma)
  any white space (tab or space)
- Class, Object, Bus, or Node delimiter: **.** (period)
- Keyword / value separator: **=**
- Continuation of previous line: **~** (More)
- Comment line: ***//***
- In-line comment: **!**
- Query a property: **?**

# Array and Matrix Parameters

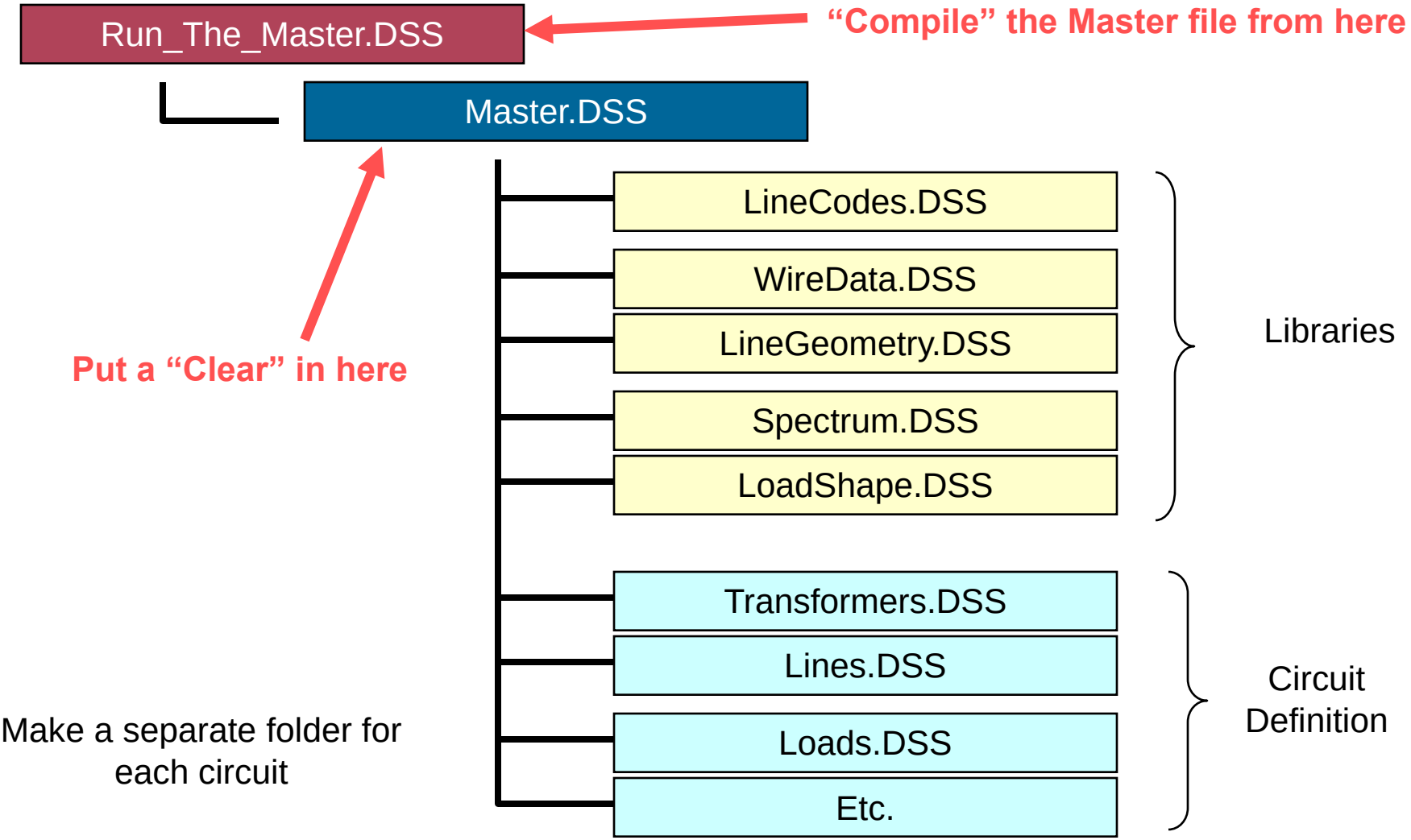
- Array
  - **kvs = [115, 6.6, 22]**
  - **kvas=[20000  16000 16000]**

- Matrix
  - ***(3x3 matrix)***
    - **Xmatrix=[1.2  .3  .3 | .3  1.2  3 | .3  .3  1.2]**
  - ***(3x3 matrix – lower triangle)***
    - **Xmatrix=[ 1.2  | .3 1.2  | .3  .3  1.2 ]**

# Specifying Bus Connections

- Shorthand (implicit)
  - `New Load.LOAD1 Bus1=LOADBUS`
  - Assumes standard 3-phase connection by default
- Explicit
  - `New Load.LOAD1 Bus1=LOADBUS.1.2.3`
    - Explicitly defines which node
  - `New Load.1-PHASELOAD Phases=1 Bus1=LOADBUS.2`
    - Connects 1-phase load to phase 2
- Default Bus template (defaults to grounded Wye)
  - `... LOADBUS.1.2.3.0.0.0.0.0.0.0.` (*ad infinitum*)
- Ungrounded-Wye Specification
  - `Bus1=LOADBUS.1.2.3.4` **(or some other unused Node number)**

# Common Sense Structuring of Script Files



ELECTRIC POWER RESEARCH INSTITUTE

The Distribution System Simulator™ (DSS)

# Solution Modes

# Distribution System Analysis Tools

- DSS has the basic tools for Planning built in:
  - Power Flow
  - Short Circuit Calculations
- In Addition, it has Several Advanced Capabilities
  - “Dynamic” Power Flow
  - Other power flow modes
  - Dynamics
  - Harmonics
- If it is not built in, you can drive it from another program such as Matlab
  - For example: Reliability Analysis

# Classes of Solution Modes

- Power Flow
  - Snapshot
  - Direct
- Dynamic Power Flow
  - Daily
  - Yearly
  - DutyCycle
  - Peakday
- Dynamics
- Harmonics
- Other Power Flow
  - LD1
  - LD2
  - Monte Carlo
    - M1
    - M2
    - M3
- Short Circuit
  - Faultstudy
  - MF - Monte Carlo Fault

EPRI | ELECTRIC POWER RESEARCH INSTITUTE

# Power Flow Modes

# Snapshot Mode

- This is the DEFAULT MODE
- Does one power flow solution at the present load level
  - Controlmode is set to “static”
  - All Control devices execute in sequence of their time delays – shortest first
    - Next control action may then be cancelled
  - You can change the default control mode if driving the DSS externally
  - You have to explicitly tell monitoring devices to “sample”
- Watch the SUMMARY window for lack of convergence

# Bus List in DSS

- The Bus List in the DSS is NOT FORMED until you do something requiring a solution or explicitly request that it be formed:
  - Solve
  - CalcVoltagebases  (zero-load power flow)
  - MakeBusList   (explicitly forms the bus list)
- If you do something that adds a bus after you do a Solve-related command the bus list is NOT automatically updated!

# CalcVoltageBases

- This command was implemented to avoid having to specify base voltages for each bus
  - You can do that by "`Setkvbase bus=… kvln|kvll =…`"
- Solves a ZERO LOAD SNAPSHOT power flow
- Set voltage bases = closest value in the set defined by
  - `Set Voltagebases=[115  12.47  0.48 …]`
- Note: this will not always work if you have two voltage bases really close together such as 12.0 and 12.47
- Remember: **The DSS works in Volts, Amps, and Ohms**
  - Voltage bases are provided for convenience

# Direct Solution Mode

- Solve System Y matrix directly
  - **I = [Y]V** (assume nominal I value)
  - No iterations
  - No compensation currents from load and generator models
    - Primitive admittance should reflect load
- Load updating forced
- Rebuilds System Y if necessary
- All Voltage and Current sources accounted for
  - Including generators if in dynamics or harmonics mode

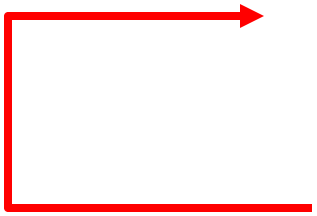

# “Dynamic” Power Flow Modes

# Daily

- Does 24 hour solution following load shape defined as “Daily”
  - See Load definition
  - There is a default loadshape defined called “default”
  - Note: the LOAD model default is NONE
- You may change the default values
  - To use a 15-min load shape,
  - Set NUMBER=96 and stepsize=(3600 4 /) … 900 s
- Meters and Monitors are reset when entering mode
  - Are not reset until mode is changed
  - Automatically sampled at end of each power flow solution
- Static control mode
  - Be careful specifying time delays! If all the same, they will all try to operate at once!

# Daily Solution Algorithm

For Number of solutions specified (set number =…)



t = t + stepsize

Solve snapshot

Sample meters and monitors

Finally,

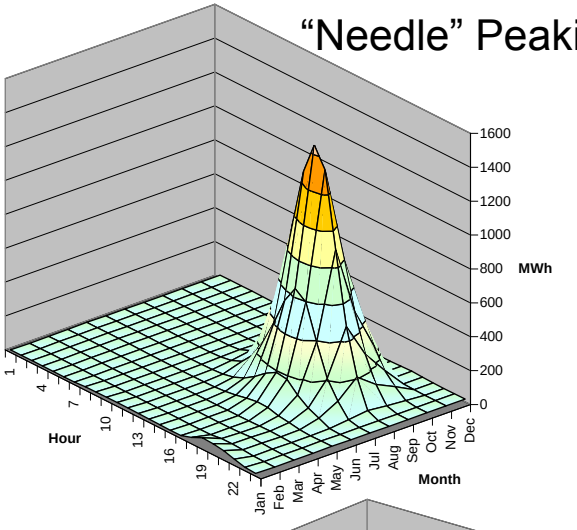
Save meters and monitors (does not reset them)

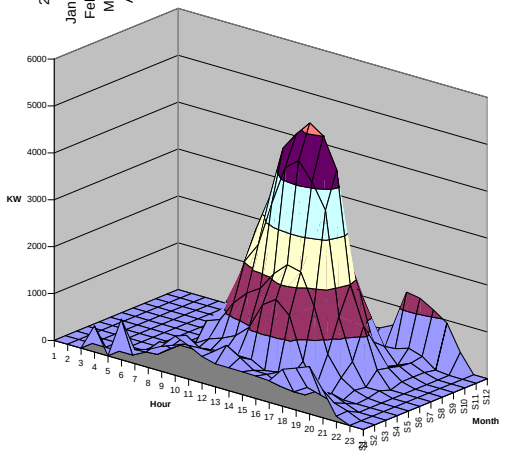
(Energymeter demand intervals may be used)

# Yearly

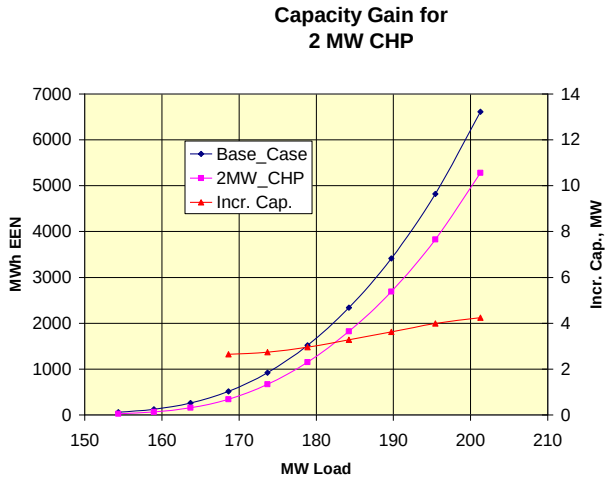
- Similar to Daily
- Defaults to
  - 1 hr stepsize (3600 s)
  - Number = 8760
- Adjusts load for growth (Set Year = …)
- Load uses loadshape defined for Yearly=…
  - Defaults to Daily … repeats over and over
  - If Daily is NONE, then load is constant
- Note: Energymeters stay open; (Monitors are saved)
  - Have to be explicitly closed by Reset or Set Year=

# Using DSS to Determine Incremental Capacity of DG




Broad Summer Peaking System




"How much more power can be served at the same risk of unserved energy?"

# Energy Meter Registers (As of Jan 2009)

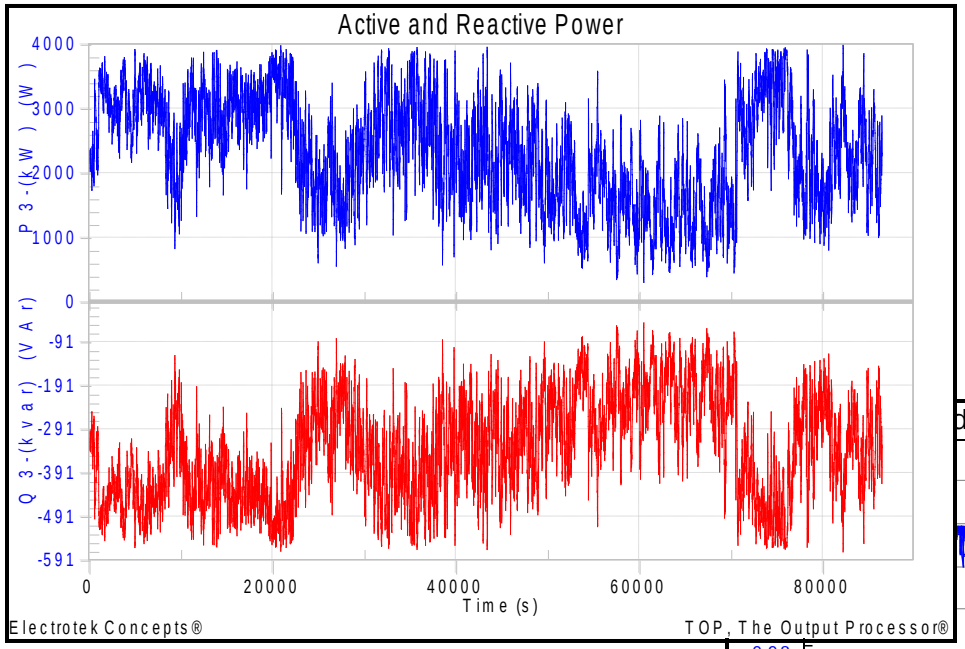
Reg 1 = kWh
Reg 2 = kvarh
Reg 3 = Max kW
Reg 4 = Max kVA
Reg 5 = Zone kWh
Reg 6 = Zone kvarh
Reg 7 = Zone Max kW
Reg 8 = Zone Max kVA
Reg 9 = Overload kWh Normal
Reg 10 = Overload kWh Emerg
Reg 11 = Load EEN
Reg 12 = Load UE
Reg 13 = Zone Losses kWh
Reg 14 = Zone Losses kvarh
Reg 15 = Zone Max kW Losses
Reg 16 = Zone Max kvar Losses
Reg 17 = Load Losses kWh
Reg 18 = Load Losses kvarh
Reg 19 = No Load Losses kWh

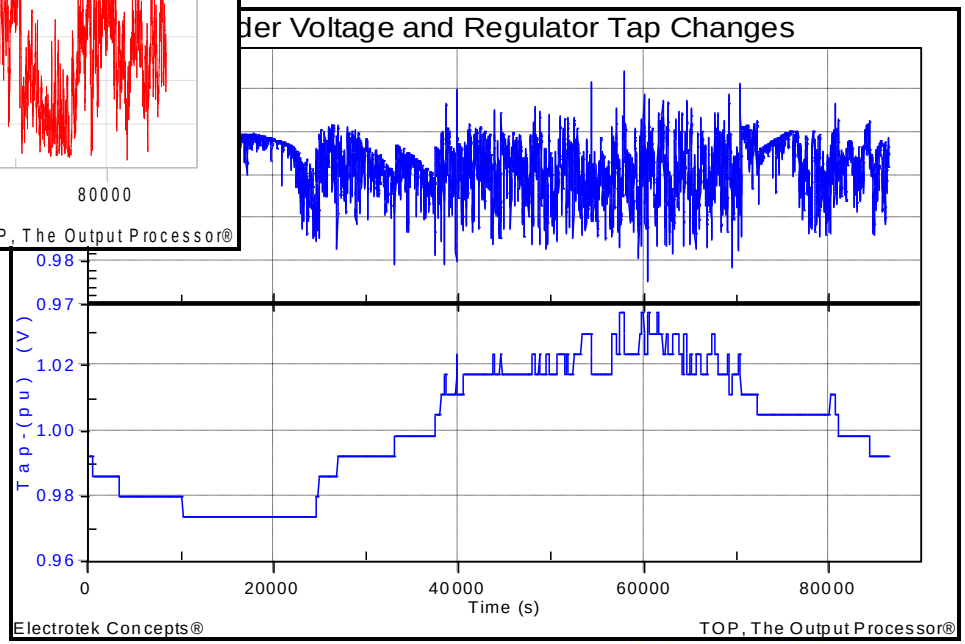
Reg 20 = No Load Losses kvarh
Reg 21 = Max kW Load Losses
Reg 22 = Max kW No Load Losses
Reg 23 = Line Losses
Reg 24 = Transformer Losses
Reg 25 = Line Mode Line Losses
Reg 26 = Zero Mode Line Losses
Reg 27 = 3-phase Line losses
Reg 28 = 1-and 2-phase Line Losses
Reg 29 = Gen kWh
Reg 30 = Gen kvarh
Reg 31 = Gen Max kW
Reg 32 = Gen Max kVA
Reg 33 = 34.5 kV Losses
Reg 34 = 0.208 kV Losses
Reg 35 = 4.16 kV Losses
Reg 36 = 230 kV Losses
Reg 37 = Aux5
Reg 38 = Aux6
Reg 39 = Aux7

# Dutycycle

- Meant to study
  - Rock crushers
  - Wind turbines
  - Rolling mills and other cyclic large motor loads
- Loads follow shape defined by Duty property
- Monitors are sampled & saved
- Energymeters are NOT sampled during the loop

# Wind Plant 1-s Simulation



EPRI | ELECTRIC POWER RESEARCH INSTITUTE

# Short Circuit Modes

# In a Power Flow Mode

- The DSS will allow you to put a FAULT object on the circuit at almost any time, even for a SNAPSHOT solution
  - Be sure to check for convergence
  - Answer may differ slightly from Faultstudy mode
- Safer (for accuracy) procedure:
  - Solve snapshot
  - Set mode=dynamics number=1 stepsize=.00001 (small)
  - Add/Enable Fault object
  - Solve  (does a direct solution with generators converted to Thevenin equivalent)

# FaultStudy Mode

- Algorithm
  - Disable all FAULT objects defined in the circuit
  - Sets Loadmodel=Admittance
  - Does DIRECT solution
    - Generators included (Thevenin equivalent)
    - Save Open-circuit voltages, Voc
  - Computes Ysc matrix at each bus
  - Computes Isc by applying
    - Isc = Ysc*Voc
  - Answer could differ slightly from specific fault solution
- Show Fault to see answers

# MF – Monte Carlo Fault Mode

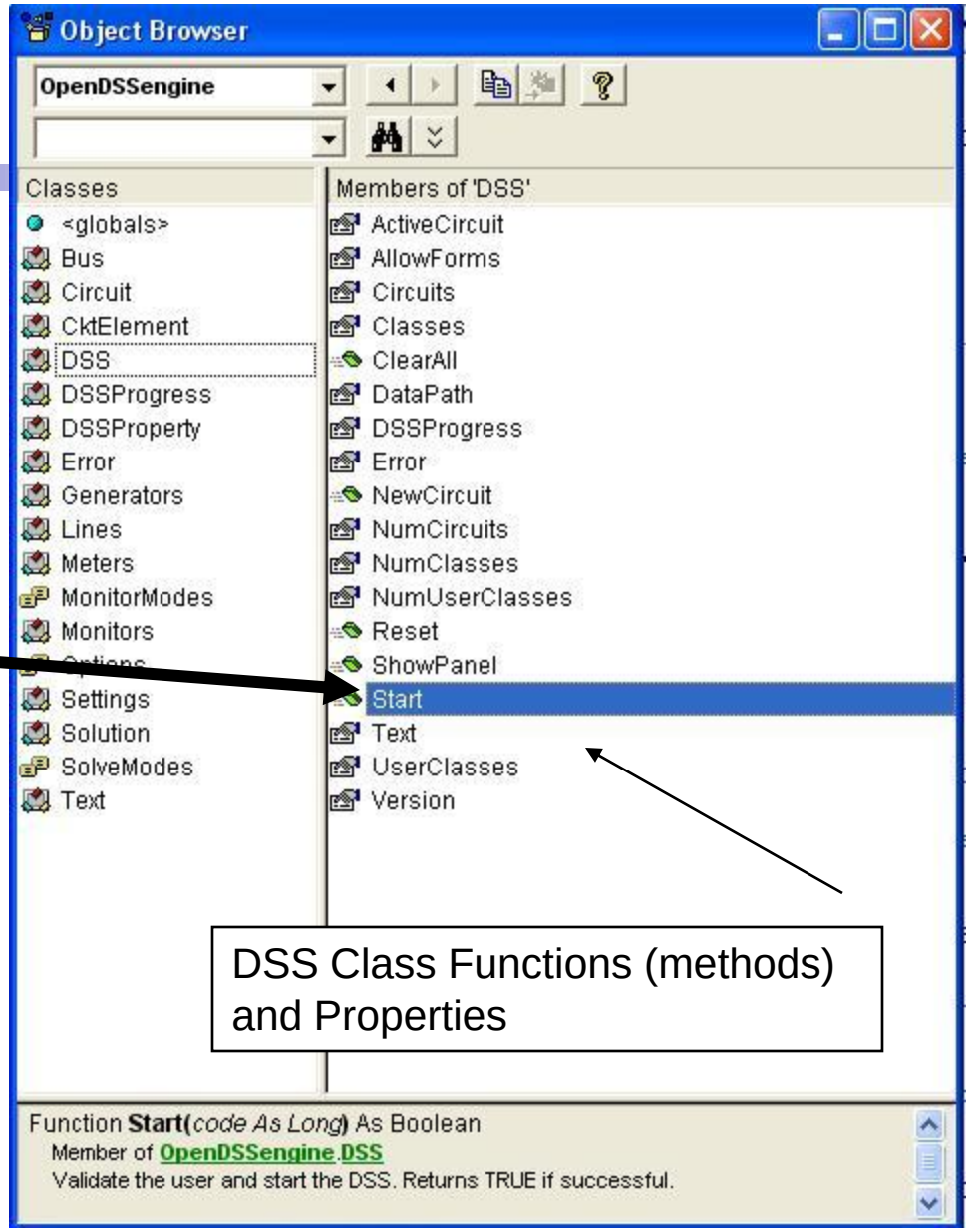
- First, define FAULT objects at all buses of interest
  - All fault types of interest
- Changes loadmodel to ADMITTANCE
- Algorithm
  - For the number of times specified
  - Picks one fault at random; disables the others
  - Randomizes the fault resistance
    - Uniform
    - Gaussian
    - Log-Normal


# Introduction to Driving the COM Server from another Application

# DSS Interface

This interface is instantiated upon loading OpenDSSEngine.DSS and then instantiates all other interfaces

Call the Start(0) method to initialize the DSS



# Instantiate the DSS Interface and Attempt Start

```
Public Sub StartDSS()

' Create a new instance of the DSS
    Set DSSobj = New OpenDSSengine.DSS
' Start the DSS
    If Not DSSobj.Start(0) Then
        MsgBox "DSS Failed to Start"
    Else
        MsgBox "DSS Started successfully"
      ' Assign a variable to the Text interface for easier access
        Set DSSText = DSSobj.Text
    End If


End Sub
```

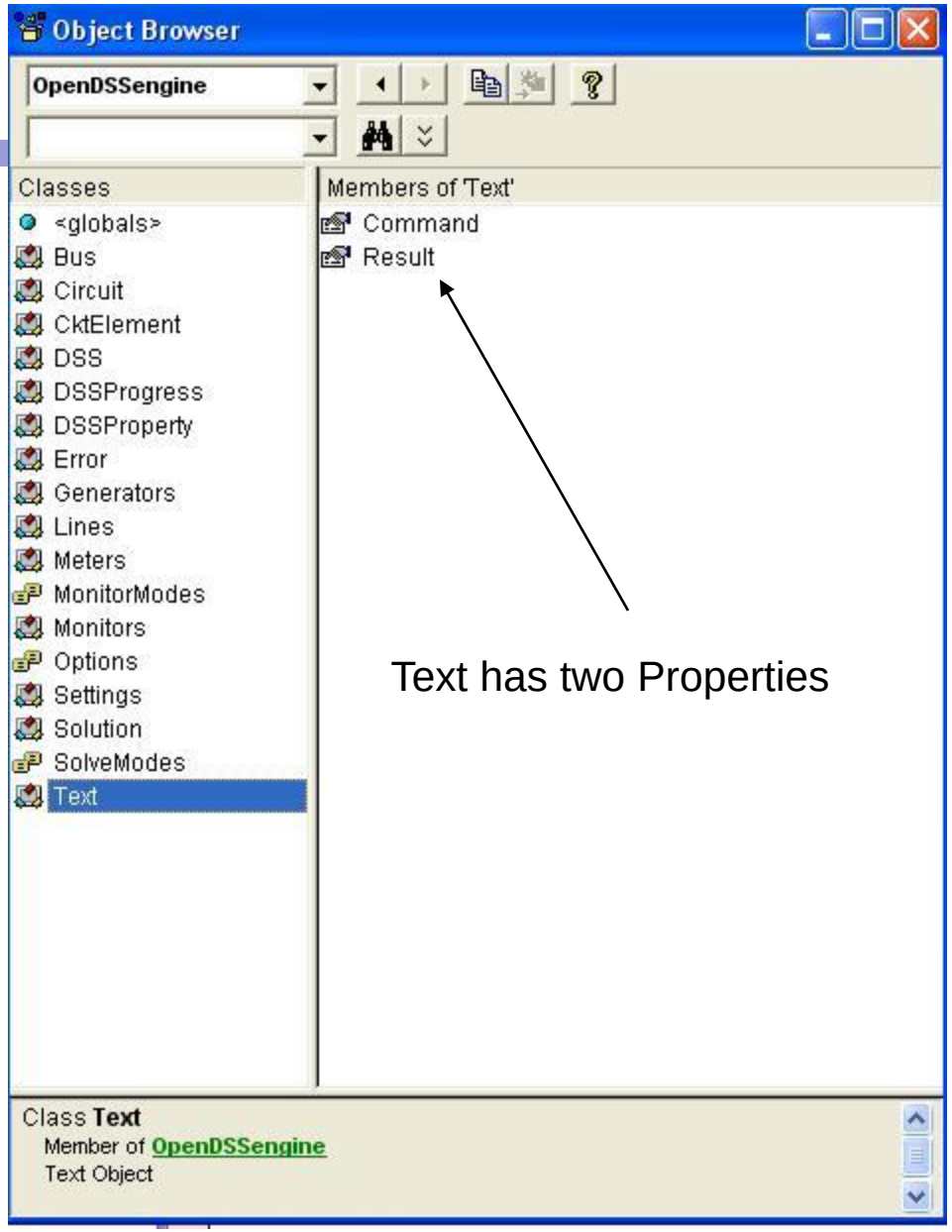
# Assign a Variable to the Text Interface

```
Public Sub StartDSS()


' Create a new instance of the DSS
    Set DSSobj = New OpenDSSengine.DSS
' Start the DSS
    If Not DSSobj.Start(0) Then
        MsgBox "DSS Failed to Start"
    Else
        MsgBox "DSS Started successfully"
      ' Assign a variable to the Text interface for easier access
        Set DSSText = DSSobj.Text
    End If



End Sub
```

# COM Interface

Interfaces as Exposed by VBA Object Browser in MS Excel

Text interface is simplest




Text has two Properties

# Result Property

- The Result property is a Read Only property that contains any result messages the most recent command may have issued.
  - Error messages
  - Requested values

```
' Example: Query line length
DSSText.Command = "? Line.L1.Length"
S = DSSText.Result       ' Get the answer
MsgBox S                 ' Display the answer
```

# Running OpenDSS From Matlab

- Starting the DSS

```
%Start up the DSS
[DSSStartOK, DSSObj, DSSText] = DSSStartup(myDir);
```

```
function [Start,Obj,Text] = DSSStartup(mydir)
% Function for starting up the DSS
% make sure we are in the proper directory
cd(mydir);
%
%instantiate the DSS Object
Obj = actxserver('OpenDSSEngine.DSS');
%
%Start the DSS.   Only needs to be executed the first time w/in a
%Matlab session
Start = Obj.Start(0);

% Define the text interface
Text = Obj.Text;
```

# Using the DSS through the DSSText Interface from Matlab (harmonics example)

```
%Compile the DSS circuit script
DSSText.Command = 'compile master.dss';

% get an interface to the active circuit called "DSSCircuit"
DSSCircuit = DSSObj.ActiveCircuit;

%Determine which connection type for the source and call
%appropriate DSS file
switch XFMRType
case 1
 DSSText.Command = 'redirect directconnectsource.DSS';
case 2
 DSSText.Command = 'redirect deltadelta.DSS';
case 3
 DSSText.Command = 'redirect deltawye.DSS';
otherwise
 disp('Unknown source Connection Type')
end

%Set the system frequency and vsource frequency for harmonic requested
DSSText.Command = ['set frequency=(' num2str(Freq) ' 60 *)'];
DSSText.Command = ['vsource.source.frequency=(' num2str(Freq) ' 60 *)'];
```

# Using the DSS through the DSSText Interface from Matlab (harmonics example) (cont'd)

```
% Vary the parameters according to a random distribution
% If more parameters need to be varied, just add them to the below
% list.  Set ParamNum to total number of parameters varied
 ParamNum = 6;    %ParamNum used for sorting/plotting
 for Case_Count = 1:Max_Cases
%Create index in the OutputData matrix to keep the cases in order
 OutputData(Case_Count,1) = Case_Count;
 % Generate random new coordinates for each conductor
 [x1 y1 x2 y2 x3 y3 geomean] = RandomGeometry(8,0.75,30);
            (... etc. etc. )
%define a new line geometry with random spacing
DSSText.Command = ['New LineGeometry.OHMOD nconds=3 nphases=3  cond=1
wire=acsr336    x=' num2str(x1) '   ' num2str(y1) '   units=ft  cond=2
wire=acsr336    x=' num2str(x2) '   ' num2str(y2) '   units=ft cond=3
wire=acsr336    x=' num2str(x3) '   ' num2str(y3) '   units=ft'];
%Solve the circuit
 DSSText.Command = 'solve';
              (etc. etc.)
```

EPRI | ELECTRIC POWER RESEARCH INSTITUTE

# Questions?