



簡明用戶指導
**The Open Distribution System Simulator™
(OpenDSS)**

作者: Jason Sexauer, OpenDSS User

翻譯: 李慧娟

License

Copyright (c) 2008-2012, Electric Power Research Institute, Inc.

All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

Neither the name of the Electric Power Research Institute, Inc., nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY Electric Power Research Institute, Inc., "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL Electric Power Research Institute, Inc., BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

目錄

LICENSE	3
目錄	5
图形	ERROR! BOOKMARK NOT DEFINED.
什麼是 OPENDSS?	8
OPENDSS 的電路模型	10
使用 OPENDSS 的圖形用戶界面(GUI)	13
用戶界面	14
工作流程	14
OPENDSS 編程基礎	18
命令語法	18
命令動詞	18
COMMENTS 註釋	20
多行命令 MULTI-LINE COMMANDS	20
包含外部文件	20
工作流程	20
代碼範例	23
COM 接口介紹	25
開始使用 COM 接口	25
TEXT 接口介紹	26
電路接口 (CIRCUIT INTERFACE) 介紹	27
算法和解的接口 (SOLUTION INTERFACE) 介紹	27
VISUAL BASIC 例子	29
MATLAB 例子	31
PYTHON 例子	33
其他資源	35
下一步該去哪兒?	35
參考資源	37

圖形

图 1: 母線 10

图 2: 電力傳輸元件 11

图 3: 电力转换元件 11

图 4. OPENDSS 用戶界面 14

图 5. 顯示線路的電壓..... 16

图 6. 電壓 剪影（VOLTAGE PROFILE） 16

图 7. 線路損失 17

图 8. 電路示範..... 23

什麼是 OpenDSS?

簡介

The Open Distribution System Simulator (OpenDSS 或 DSS) 是一個全面的電力系統配電網仿真工具。OpenDSS 是開放源代碼的配電網仿真計算分析軟件。它既可以作為一個獨立的執行程序使用，又可以作為動態鏈接庫（DLL）被很多其他軟件平臺調用。作為獨立執行程序使用時，OpenDSS 有一個文本用戶界面供使用者建立基於文本的模型和察看仿真計算結果。

這個軟件支持幾乎所有常用的用於配網規劃分析的基於有效值（RMS）的穩態分析（即頻域分析）。並且它還支持許多有關智能電網的新分析來迎接電網分析的未來需要。這個軟件的特性包括：支持分布式發電分析；支持電能傳送的效率分析；智能電網應用；諧波分析。這個軟件設計的有良好的可擴充性，所以可進行修改升級以適應未來需求。

DSS 的另一個特點是它可以進行連續時間的系統仿真計算, 比如，它可以仿真計算一個電路在一年的情況。這個軟件內嵌仿真計算模式，比如：

- 幀計算 (snapshot)
- 天計算(daily)
- 年計算(yearly)
- 諧波分析 (harmonic)
- 動態分析(dynamic)
- 故障（短路）分析(fault)
- Monte Carlo 故障分析 (M1,M2,M3)
- 等等 ...

這些模式因開發者在不同研究項目中需要而被先後加入這個軟件。然而，開發者並不能預測使用者的所有需求，所以這個軟件還提供了組件對象模型（Component Object Model (COM)）接口，DSS 作為 DLL，讓用戶編寫其他程序調用以滿足不同的需要。通過 COM 接口，用戶可以用其他程序設計執行不同的客戶解決方案和執行仿真計算，包括可以通過這種方式定義系統模型。這樣，DSS 可以獨立於任何的數據庫或者固定的文本電路模型定義。例如，它可以在 MS Office 中用 VBA 驅動，或者其他有 COM 功能的第三方分析軟件驅動。用戶很普遍的用 Mathworks MATLAB，Python, C#, R, 還有其他軟件語言來驅動 DSS。這個特性提供了一個強大的外部分析功能和很好的結果圖形展示。

簡單的歷史

最早 OpenDSS 的開發始於 1997 年四月在 Electrotek Concepts, Inc. 當時這個程序被稱為“DSS”，配網仿真計算器。Roger Dugan 是主要開發者，隨後 Tom McDermott 加入。他們倆組成了開發團隊直至 2001 年 Tom 離開 Electrotek。Roger 繼續維護升級這個程序到最近 Tom 通過 OpenDSS 這個項目又重新回到開發團隊。EPRI Solutions 在 2004 年獲得了這個軟件，在 2007 年它又與 EPRI 合並。在 2008 年，EPRI 發布了該軟件帶有開放源代碼的許可，來與其他智能電網領域正在進行的電網現代化努力相合作。

這個軟件上接近於諧波分析軟件或者動態軟件，而不是一般意義上的潮流計算軟件。關於

這點的一個最明顯的表現是母線（bus）在這個軟件中無足輕重，而在其他大部分的潮流軟件中，母線占據中心位置，其他的計算都以它為基礎。在 OpenDSS 中，母線是當需要時動態生成的。對一個將被主要當作潮流計算的工具，這最初的設計可能看起來很奇怪。但是，它賦予了這個工具很大的建模靈活，尤其是它能適應各種不同的負荷模型和特殊的電路結構。用一個諧波潮流工具解正常潮流是容易的，反之則不然。

這個文檔的目的

這個文檔希望給新用戶一個快速指導。它包括很多 OpenDSS 手冊上的信息，但它並不全面深度覆蓋這個軟件的使用。有興趣的用戶可以參閱文中列出的其他資料以獲得更多信息。

OpenDSS 的電路模型

OpenDSS 包含一個基於有效值（RMS）的穩態配網模型和一個連接這些電網元件控制的內部通訊系統。最基本的電路模塊是電力傳輸元件，比如傳輸線，變壓器，電容器，和電力轉換元件，例如發電機和負荷。輔助模塊，例如控制，發電或負荷曲線形狀，計量器，參數提取，它們可以進一步細化電力傳輸和轉換元件的定義。更多這些模塊的信息可以在用戶手冊中發現。通過這些元件，代表一個相連系統的母線和節點會被動態生成，這和傳統的以母線為中心的潮流計算工具根本的不同。

母線

母線可以有 [1..N]個節點。母線是電路元件的連接點。在很多電力系統分析軟件中，母線和節點基本是同一概念，但是它們在 OpenDSS 中是完全不同的。母線包含節點，就是說，一個母線可以有多個節點。

Node 0 總是被作為一個母線的參考節點/接地節點/0 V 節點。其余節點傳統上是其他相，如，node 2 是 B 相。當指定一個母線的位置時，包含的節點要被包括。（node 0 除外，它被認為是一直存在）例如，指定一個 3 相母線 BUSNAME，

BUSNAME.1.2.3

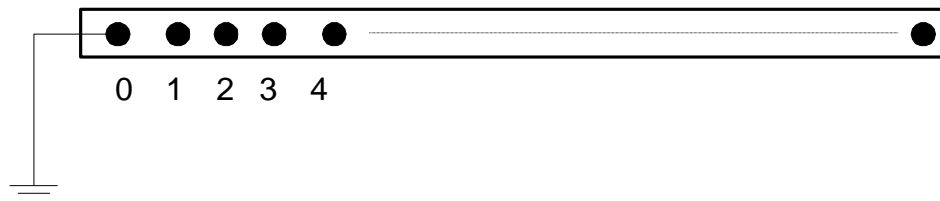


图 1: 母線

電力傳輸元件

電力傳輸(PD)元件 通常是由兩個或多個終端組成。它們的基本功能是從一點向其他點運送電能。電力系統中，最普通的電力傳輸元件是線路和變壓器。所以它們一般含有多個終端（電容器和電感器當並聯入電網而不是串聯時是個例外）。

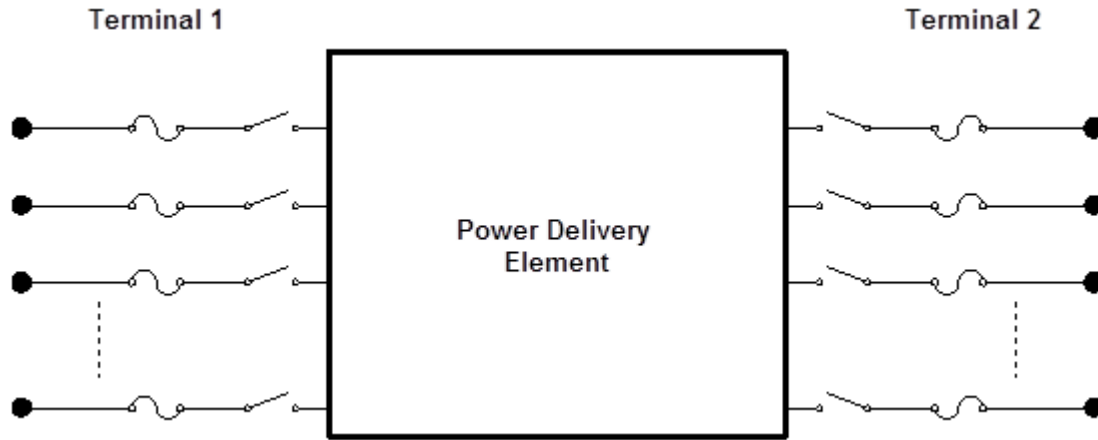


图 2: 電力傳輸元件

電力轉換元件

電力轉換 (PC) 元件把電能轉換為其他能量或反之。有些可能短時存儲能量再提供出來，比如一些無功 (reactive power) 元件。大部分只有一個電網連接點，所以一個多相的終端。最普通的電力裝換元件是發電機和負荷。

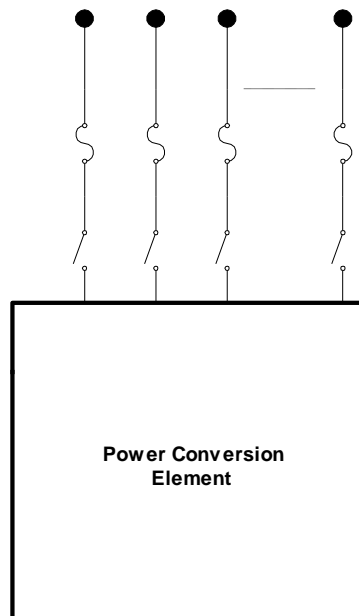


图 3: 电力转换元件

輔助元件

輔助元件可以進一步附加在電力傳輸和轉換元件上。輔助元件功能包括：方便的提取系統數據參數，提供控制功能，提供測量，或提供相應的曲線信息以便時間段仿真計算使用。最常用的有：

- **LineCode** - 定義線路的類型，比如線路的阻抗。
- **LineGeometry, LineSpacing, and WireData** - 一起使用，它們共同定義線路的類型基於一些最基本的物理參數，如線路的幾何平均半徑（ Geometric Mean Radius (GMR) ）。
- **LoadShape** - 定義負荷曲線的相乘因子或實際 kw/kvar 以用於時間段仿真計算
- **Spectrum** - 定義一個電力傳輸元件發出的諧波的頻譜
- **EnergyMeter** - 用於收集一條饋線上的不同計量信息
- **Monitor** - 用於收集具體元件的潮流計算結果
- **CapControl and RegControl** - 模擬可控電容器和可控線路電壓調壓器的控制

使用 OpenDSS 的圖形用戶界面(GUI)

OpenDSS 包含一個圖形用戶界面(GUI)，它提供了一個編寫和分析電力系統案例的結構化的環境。圖形用戶界面(GUI) 是用戶使用 OpenDSS 仿真計算引擎的兩種基本方式的一種，另一種是通過組建對象模型（COM）接口。在隨後將會介紹這種方式，更多的細節在 OpenDSS 用戶手冊（OpenDSS Manual）。

應該指出的是 GUI 更是一個輔助電路分析生成，調試代碼的工具,而不是取代代碼編程。用戶傳輸信息給 DSS 根本上是通過傳送給 OpenDSS 命令處理器的文本來實現的。在 DSS 的 GUI 中，代碼是這樣執行的：

1. 選中要執行的代碼
2. 用戶然後右擊鼠標選擇“Do Selected”這個選項,它的快捷鍵組合是 Ctrl-D.
3. 選擇的代碼也可以通過“Do”菜單或者這個菜單下的快速按鈕.

所有通過 GUI 執行的命令都有對等的 OpenDSS 編程命令. 這些命令都詳細的記錄在 OpenDSS Manual 中。並且這些命令和元件屬性都可以在用戶界面的“help” menu 找到。你也可以使用“Edit” menu 下的“Record Script”工具，把在 GUI 中執行的命令記錄到一個 dss 文件中。

仿真計算的結果可以存儲在 CSV 文件中。OpenDSS 提供幾個標準格式的文本報告文件（參考 Show 和 Export 命令）。用戶如果需要更複雜的報告文件，可以使用 Excel 或者其他的應用程序通過 COM 接口來控制 OpenDSS 完成。（隨後將介紹 COM 接口）。

用戶界面

當打開 OpenDSS 時，你首先看到的是下面的界面：

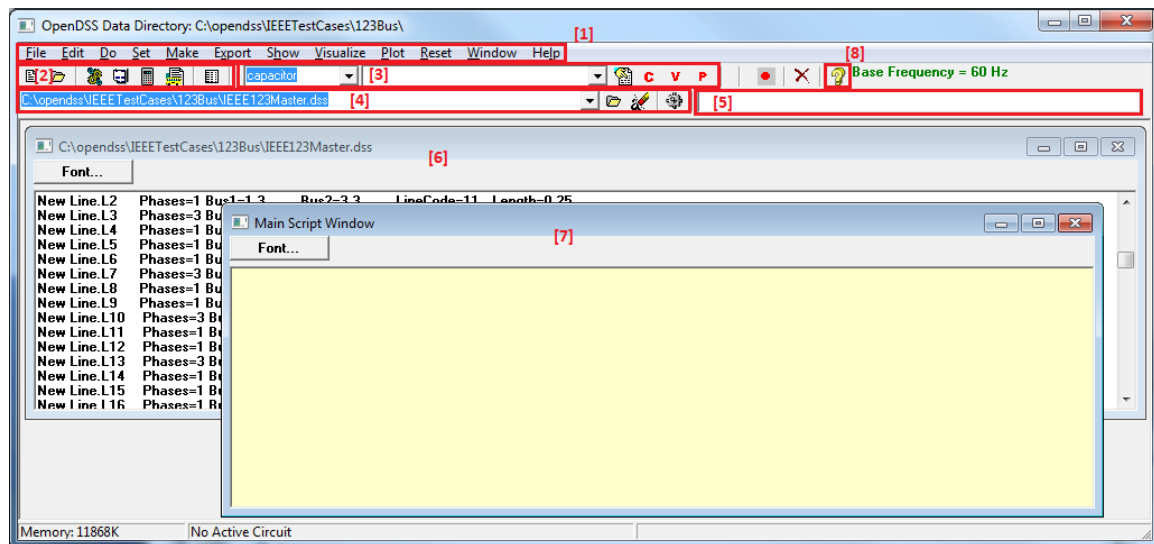





圖 4. OpenDSS 用戶界面

主要組件包括：（序號和圖 4 號碼相對應）

1. **菜單結構**，它來驅動 OpenDSS 中主要的工作流。菜單包括：
 - **Set 菜單**，這裏你可以設置仿真計算參數，這些參數也可以通過編程設置 options 中的參數 來完成
 - **Export 菜單**，這個命令用來保存各種輸出報告到 CSV 文件。
 - **Show 菜單**，它和 Export 菜單命令輸出的文件信息基本相同，不同的是它直接在 GUI 中顯示報告結果。
 - **Visualize 菜單**，它顯示選中元件（圖 4 中的“3”）的圖像結果。
 - **Plot 菜單**，它提供有關仿真計算電路的圖像結果。
2. **工具條**，這裏你可以直接使用很多常用的 OpenDSS 命令，比如“Solve,” “Summary,” 和 “Do Command.”
3. **元件工具**，這裏你可以選擇電路元件以便進行編輯或者顯示它的圖像結果
4. **編碼工具**，這裏你可以從已打開的代碼文件選擇要執行的文件。
5. **結果欄**，這是個縮小版的 result widow。Result widow 可以通過 show 菜單下的 Result Form 顯示。
6. **代碼 Windows**，這裏顯示電路模型的 dss 文件。
7. **主代碼 Window** 是 OpenDSS 的交互窗口。這裏用戶可以鍵入命令行，然後通過選中它們，用“Do Command”的快捷組合 (Ctrl-D) 來執行。這裏輸入的命令會被保留在窗口中。
8. **Help 按鈕**，這裏你可以看到 OpenDSS 的命令和元件屬性的幫助參考信息

工作流程

一般來說，用戶需要按以下步驟：

1. **定義電路：** 定義線路，變壓器，負荷，發電機，等…
 - a. 定義電路是通過編寫 dss 文件實現。下一章將介紹更過關於編程的信息。
 - b. 一旦代碼編寫完成，可以使用  按鈕運行選定的代碼。
2. **設定電路的仿真計算選項，** 比如，仿真計算模式 (snapshot, daily, harmonic, 等) 這可以通過 “Set” 菜單的命令來實現。最基本的仿真計算模式是 幀計算 (snapshot)，它類似於傳統的潮流計算。關於更多的電路仿真計算選項，參閱 OpenDSS 手冊。
3. **解潮流計算問題**
 - a. 首先，通過運行 Do 菜單下的 “Calc Voltage bases” 來確保生成母線節點列表和尋找到基本電壓。
 - b. 然後使用  按鈕來解電路。
4. **分析已解電路** - 不同的分析包括不同的使用細節。但是一些通常的任務包括：
 - a. 檢查線路，變壓器，負荷，等元件 - 首先，從元件工具中選定元件類型然後元件，使用 C, V, or P 按鈕來查看電流，電壓和功率。使用  按鈕來編輯元件屬性。圖 5 顯示一個查看線路電壓的例子。
 - b. 圖形化顯示電壓-距離 - 在主程序窗口中鍵入 plot profile，並選中它然後使用 Ctrl-D 組合鍵來執行這個命令。Voltage profile 將顯示沿著這個電路離開變電站電壓如何變化; 圖 6 給出了一個例子。Plot 命令還有其他參數，參閱 OpenDSS Manual 以獲得更多詳情。
 - c. 在線路連接圖上 (one-line diagram) 圖形化顯示數據 - 如果你通過 buscoords 命令提供了母線的位置信息 (參閱 OpenDSS Manual)，已可以在這個電路的地圖 (one-line diagram) 上顯示潮流數據。在菜單上選擇 Plot > Circuit Plots > Circuit Plot. 圖 7 顯示了一個例子。
 - d. 輸出數據以使用其他軟件分析 - 所有 OpenDSS 的結果可以通過在 “Export” 菜單中不同的命令來輸出。結果存儲為 .csv 文件

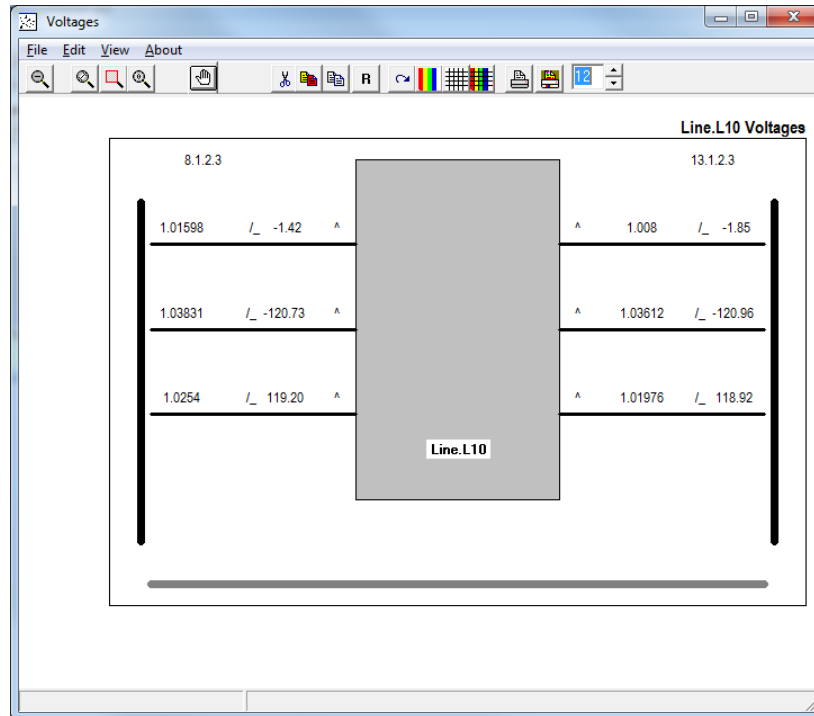


图 5. 顯示線路的電壓

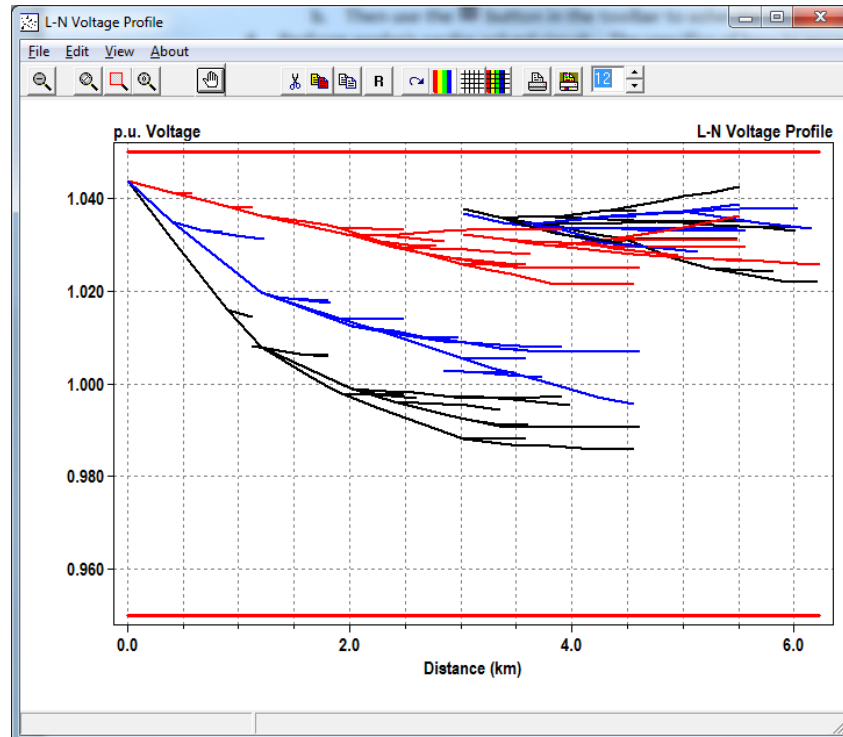


图 6. 電壓 剪影 (Voltage Profile)

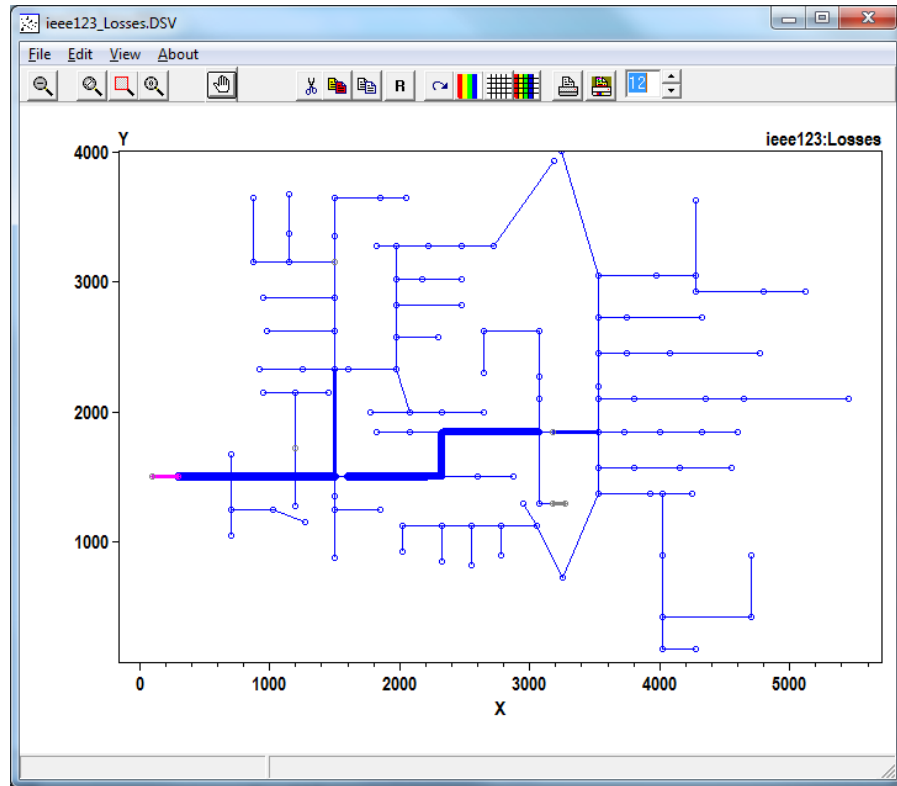


图 7. 線路損失


OpenDSS 編程基礎

DSS 的所有功能都能用文本編程命令來實現。這些文本可以通過以下方式:

1. 在程序窗口選擇並執行,
2. 通過 COM 接口,
3. 標準文本文件, 通過 **Compile or Redirect** 命令

這些方式令使用 DSS 很方便, 無論是對一個小型簡單的電路分析還是一個複雜的分析。這同時也為使用其他軟件的用戶提供了便利。

OpenDSS 的編程語言是區分大小寫的 (case insensitive) 。

請查看 **Help** 命令來獲得當前最新的命令和屬性 (通過菜單上的  button)。

命令語法

命令是一個單行的編程文本. 有些命令動詞直接作用於電路元件(比如 **New and Edit** 命令)。這類型的命令 格式如下:

```
CommandVerb (命令動詞) ElementClass.Element.Param1=Val1, Param2=Val2
```

有的命令動詞不直接作用於電路元件 (比如 **Plot** 和 **Export** 命令)。這類型的命令有如下格式:

```
CommandVerb Param1=Val1, Param2='Value 2', Param3=(1 2 3)
```

命令動詞

一個完整的命令動詞列表可以通過界面上的  按鈕獲得。一些常用命令動詞包括:

- **New** - 生成新的電路元件。
- **Edit** - 編輯指定的電路元件
- **Set** - 定義仿真計算的參數, 比如 **mode** (仿真計算模式) . 參見 “options” 在命令幫助文件中
- **Solve** - 在當前電路上進行仿真計算
- **Show** - 用文本顯示計算的潮流結果。
- **Export** - 輸出計算的潮流結果到 **text** 或者 **csv** 文件。
- **Plot** - 圖形化潮流結果。

如果沒有指明具體命令, 默認 **Edit** 命令

參數

參數/值 可以用逗號 (,) 或者空白 (blank, tab) 分開. 如果值要求分割符的話, 下列格式可被接受。儘管實際上可以互換, 還是鼓勵遵循下面的標準格式:

- 雙引號 "..." 字符串
- 單引號 '...' 字符串
- 小括號 (...) 數組
- 中括號 [...] 數組
- 大括號 {...} 嵌入數學式子 (參閱 OpenDSS Manual)

類成員或者元件的參數（也叫屬性）的接口可以通過點 “.” 獲得。所用的元件都應通過它們的全名來指定，除非通過上下文可以推斷出它的類。全名有如下格式：

```
ElementClass.ElementName
```

元件的 屬性的全名格式如下：

```
ElementClass.ElementName.PropertyName
```

當使用同一命令作備於同一元件的不同屬性時，除第一個外，其他的屬性不需要全名。比如¹：

```
Line.L1.Bus1=1, Bus2=5
```

除了指定屬性然後賦值外，屬性賦值還可以按默認順序進行。按默認順序和指定屬性賦值可以混合使用，例如：

```
New EnergyMeter.Feeder Line.L115, terminal=1, enabled=false
```

當生成一個新元件時，許多屬性有默認值。在幫助文件中有完整的記錄有關於什麼屬性有默認值及默認值是多少，什麼必須被指定。一般來說 element, object, terminal, bus1, 和 bus2 屬性沒有默認值，它們必須被指定。

¹ 注意此处命令没有指定，默认是 Edit

COMMENTS 註釋

代碼的註釋可以用 // 或者 !, 如下:

```
// Edit the voltage regulator control
RegControl.Ctrl1. maxtapchange=1 ! Limit to one tap change
```

多行註釋可以通過 /* ... */. 注意 /*必須在第一列. 比如:

```
/*    comment out the next two monitors
New Monitor.Source-PQ  Vsource.source 1  mode=1 ppolar=no
New Monitor.source-VI  Vsource.source 1  mode=0 VIpolar=Yes
*/
```

多行命令 MULTI-LINE COMMANDS

如果一個命令超過一行, ~ 符號 應被使用, 比如:

```
New Line.L1  Bus1=1, Bus2=2, Length=1
~  units=mi, geometry=3PH_3/0_Horiz
```

註意, 上面的代碼實際上是兩個命令。 所以, 任何沒有默認值的屬性在生成這個元件時必須在第一行中定義。下面的代碼會發生錯誤:

```
// This will error because Bus1 and Bus2 are not set in the first line
New Line.L1  Length=1, units=mi
~  Bus1=1, Bus2=2, geometry=3PH_3/0_Horiz
```

包含外部文件

有兩種方式來包含外部文件

1. Compile 命令,它會嵌入另一個 OpenDSS 代碼在當前處.
Compile 命令會改變路徑到外部文件的路徑.
2. Redirect 命令,同樣, 它會嵌入另一個 OpenDSS 代碼在當前處.
不同的是, redirect 命令會保留原有的路徑.

一些屬性, 例如 LoadShapes 的 mult,可能會需要大量的數據, 不方便直接放入主代碼中或者來源於其他各式. File ,可以讀入一個文件並使用它的內容作為一個元件的值。比如:

```
LoadShape.LS1 mult=(File='Example.csv')
```

參閱 OpenDSS Manual 關於 File 和他的姐妹功能 sngFile and dblFile.

工作流程

前面已經提及, 任何通過 GUI 完成的任務都可通過代碼完成。下面描述的工作流程和上節的工作流程類似:

1. 定义电路 定义线路, 变压器, 负荷, 发电机, 等...

- a. 使用 `Clear` 命令来清除任何以前载入的电路
ex: `Clear`
 - b. 使用 `New c` 命令定义新电路
ex: `New object=circuit.ExampleCircuit`
 - c. 使用 `New` 命令定义新元件
ex: `New line.L1 bus1=1 bus2=2 linecode=336ACSR length=1.25`
2. 设定电路的仿真计算选项, 比如, 仿真计算模式 (snapshot, daily, harmonic, 等)
 - a. 使用 `Set` 命令
ex: `Set mode=snapshot voltagebases=(115, 12.47, .14)`
3. 解潮流计算问题
 - a. 首先, 使用 `CalcVoltageBases` 命令确保母线列表生成, 发现相对应的电压。
ex: `CalcVoltageBases`
 - b. 使用 `solve` 命令来进行仿真计算。
ex: `Solve`
4. 分析已解电路-不同的分析包括不同的使用细节。但是一些通常的任务包括:
 - a. 检查线路, 变压器, 负荷, 等元件- 使用 `Visualize` 命令
ex: `Visualize element=Line.L1 what=powers`
 - b. 图形化显示电压-距离- 使用 `plot` 命令
ex: `Plot profile`
 - c. *Visualize data onto a one-line of the feeder – Use the Buscords and Plot commands*
ex: `Buscords BusCordsFile.dat`
`Plot circuit`
 - d. 输出数据以使用其他软件分析— 使用 `Export` 命令
ex: `Export voltages`
5. 定義電路: 定義線路, 變壓器, 負荷, 發電機, 等...
 - a. 使用 `Clear` 命令來清除任何以前載入的電路
ex: `Clear`
 - b. 使用 `New c` 命令定義新電路
ex: `New object=circuit.ExampleCircuit`
 - c. 使用 `New` 命令定義新元件
ex: `New line.L1 bus1=1 bus2=2 linecode=336ACSR length=1.25`
6. 設定電路的仿真計算選項: 比如, 仿真計算模式 (snapshot, daily, harmonic, 等)
使用 `Set` 命令
ex: `Set mode=snapshot voltagebases=(115, 12.47, .14)`
7. 解潮流計算問題
 - a. 首先, 使用 `CalcVoltageBases` 命令確保母線列表生成, 發現相對應的電壓。
ex: `CalcVoltageBases`
 - b. 使用 `solve` 命令來進行仿真計算。
ex: `Solve`
8. 分析已解電路 - 不同的分析包括不同的使用細節。但是一些通常的任務包括:
 - a. 檢查線路, 變壓器, 負荷, 等元件 - 使用 `Visualize` 命令
ex: `Visualize element=Line.L1 what=powers`
 - b. 圖形化顯示電壓-距離 - 使用 `plot` 命令
ex: `Plot profile`

- c. Visualize data onto a one-line of the feeder – Use the Buscords and Plot commands
ex: Buscords BusCordsFile.dat
Plot circuit
- d. 輸出數據以使用其他軟件分析 – 使用 Export 命令
ex: Export voltages

代碼範例

以下面電路為例子進行示範:

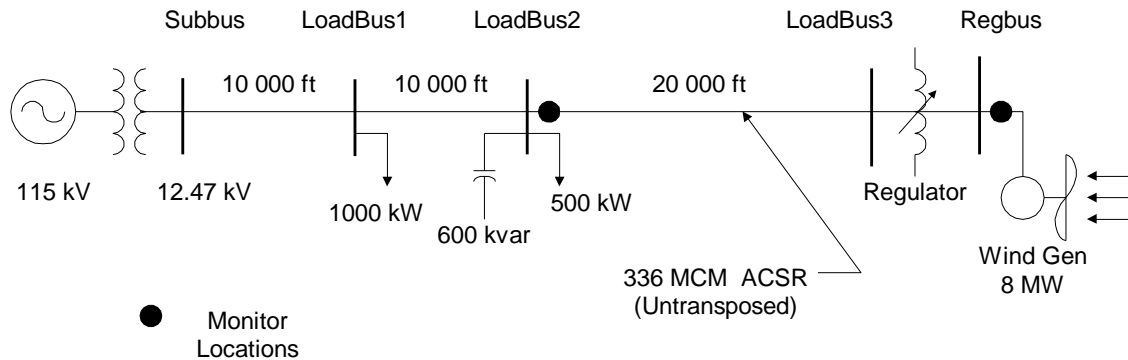


图 8. 電路示範

下面的代碼生成上述電路

```
// The first step is always to clear the DSS and instantiate a new circuit
clear
New object=circuit.ExampleCircuit
~ basekv=115 1.00 0.0 60.0 3 20000 21000 4.0 3.0 ! edit the voltage source

// Define some load shapes for the loads and wind
! This is an example of defining parameters via their default order
! in this case, the num of points and interval
New loadshape.day 8 3.0
~ mult=(.3 .36 .48 .62 .87 .95 .94 .60)
! This is an example of an inline calculation, see OpenDSS Manual for more info
New loadshape.wind 2400 {1 24 /} ! unit must be hours
~ mult=(file=zavwind.csv) action=normalize ! wind turbine characteristic

// Define a linecode for the lines - unbalanced 336 MCM ACSR connection
New linecode.336matrix nphases=3 ! horizontal flat construction
! rmatrix, xmatrix, and cmatrix are in lower triangular matrix format.
! see the OpenDSS Manul for more details on how to specify matrixes.
! In ohms per 1000 ft
~ rmatrix=(0.0868455 | 0.0298305 0.0887966 | 0.0288883 0.0298305 0.0868455)
! In ohms per 1000 ft
~ xmatrix=(0.2025449 | 0.0847210 0.1961452 | 0.0719161 0.0847210 0.2025449)
! In nf per 1000 ft
~ cmatrix=(2.74 | -0.70 2.96 | -0.34 -0.71 2.74)
~ Normamps = 400 Emergamps=600

// Define Substation transformer
! Note that the buses property provides an alternate way to specify the
! buses beyond bus1= and bus2=
New transformer.subxfm phases=3 windings=2 buses=(SourceBus subbus)
~ conns='delta wye' kvs=(115 12.47) kvas=(20000 20000) XHL=7

// Define the lines
New line.line1 bus1=subbus bus2=loadbus1 linecode=336matrix length=10
New line.line2 loadbus1 loadbus2 336matrix 10
New line.line3 Loadbus2 loadbus3 336matrix 20
```

```
// Define the loads
New load.load1 bus1=loadbus1 phases=3 kv=12.47 kw=1000.0 pf=0.88 model=1
~ class=1 duty=day
New load.load2 bus1=loadbus2 phases=3 kv=12.47 kw=500.0 pf=0.88 model=1
~ class=1 duty=day conn=delta

// Capacitor with control
New capacitor.Cap1 bus1=loadbus2 phases=3 kvar=600 kv=12.47
New capcontrol.Cap1Ctrl element=line.line3 terminal=1 capacitor=Cap1
~ type=current ctratio=1 ONsetting=60 OFFsetting=55 delay=2

// Regulated transformer to DG bus
New transformer.reg1 phases=3 windings=2
~ buses=(loadbus3 regbus)
~ conns='wye wye'
~ kvs=(12.47 12.47)
~ kvas=(8000 8000)
~ XHL=1 ! tiny reactance for a regulator

// Regulator Control definitions
New regcontrol.subxfmCtrl transformer=subxfm winding=2 vreg=125
~ band=3 ptratio=60 delay=10
New regcontrol.reg1Ctrl transformer=reg1 winding=2 vreg=122 band=3
~ ptratio=60 delay=15

// Define a wind generator of 8MW
New generator.gen1 bus1=regbus kv=12.47 kW=8000 pf=1 conn=delta
~ duty=wind Model=1

// Define some monitors so we can see what's happening
// (See documentation on how the mode parameter works)
! Monitor the power output of the wind turbine
New Monitor.gen1 element=generator.gen1 terminal=1 mode=1
! Monitor the voltage and currents at the second load bus
New Monitor.loadbus2 load.load2 1 mode=0
! Monitor sequence voltages and currents magnitudes of line 3, terminal 1
New Monitor.line3 line.line3 1 mode=48
! You need an energy meter in order to get line distances for a profile plot
New Energymeter.em1 line.line1

// Define voltage bases so voltage reports come out in per unit
Set voltagebases=(115 12.47 .48)

// Generate the bus list and figure out the voltage bases
Calcvoltagebases

// Simulation options to make the cap and reg controllers operate in sync
// with the rest of the simulation
Set controlmode=time
// Simulation options to do a time based simulation for 24 hours (86400 sec)
// with a time step of 1 sec starting at hour 0, second 0
Set mode=duty number=86400 hour=0 stepsize=1 sec=0

// Conduct the simulation
Solve

// Show some results
! Plot how the voltage at loadbus1 looked during the day
Plot monitor, object=loadbus2, Channels=(1,3,5)
! Visualize the line's flow as it appear at the last timestep
Visualize element=Line.line1 what=powers
! Show the voltage profile on the feeder as it appeared at the last timestep
Plot profile
```


COM 接口介绍

盡管在標準用戶界面中可完成很多工作，但是用戶可以通過 COM 接口獲得更多的靈活性。COM 接口可以使用戶在其他軟件中開發算法來驅動 OpenDSS 引擎以實現一些 DSS 中尚未開發的功能。比如優化算法。目前 DSS 僅有一些簡單算法,但是可以外部開發複雜算法。這些外部算法將依靠 DSS 來仿真配網的相應狀態，以此來調節優化有關控制變量。

一個使用 COM 接口的例子是循環執行代碼。OpenDSS 沒有循環執行代碼的能力。最接近的是 Next 命令，它循環增加時間。而在其他語言中，循環相對很容易實現並且運行相對較快。

詳細的 COM 接口的記錄可以在 OpenDSS manual 中找到。但是最新版的記錄是在 COM 接口中。使用一個類庫瀏覽器（type library browser (TLB)） - 如 Microsoft Office 的 Visual Basic Editor - 在 Excel 的 VBA 的編輯器中可以瀏覽 DSS 的 COM 接口所包含的庫（library）。在 Excel 的 VBA 的 Editor 中，在 reference (在 Tools 菜單中) 中填加 OpenDSSEngine. 然後你可以使用"Object Browser"（在 View 菜單中）瀏覽 OpenDSSEngine 的庫。

在這個簡介中，Visual Basic 的代碼將被作為例子，因為它相對其他編程語言較容易被讀懂並且 較易獲得。你可以使用 Microsoft Excel 的 macro editor 或者免費版的 Visual Studio Express 的 Visual Basic 來嘗試下面的例子。在 Mathwork's MATLAB 和 Python 的例子在笨重的最後也會提到。OpenDSS 用戶還有在其他編程語言成功使用 COM 接口的經驗，比如 C++, C#, and R 等。

開始使用 COM 接口

首先，必須確認你的編程環境和 OpenDSS 的 COM 接口相連。在 Visual Basic 中，是這樣實現的：Tools > References or Project > Add Reference。從這裏選擇 OpenDSSEngine COM 類型庫。

使用下面的代碼初始化一個 OpenDSS 對象並且生成一個鏈接：

```
' Declare the OpenDSS related variables
Dim DSSObj As OpenDSSEngine.DSS
Dim DSSText As OpenDSSEngine.Text
Dim DSSCircuit As OpenDSSEngine.Circuit
Dim DSSSolution As OpenDSSEngine.Solution

' Instantiate the OpenDSS Object
DSSObj = New OpenDSSEngine.DSS

' Start up the Solver
If Not DSSObj.Start(0) Then
    MsgBox("Unable to start the OpenDSS Engine")
    Return
End If

' Set up the Text, Circuit, and Solution Interfaces
DSSText = DSSObj.Text
DSSCircuit = DSSObj.ActiveCircuit
DSSSolution = DSSCircuit.Solution
```

可以看出，從 DSS parent object 中存在幾個子類（children classes）：

- Text, 它提供對命令編輯器的接口（command line interpreter interface）。使用它，你可以直接使用在前面章節中提到的 OpenDSS 命令。
- Circuit, 它提供了對組成電路元件的接口。使用它的成員，你可以編輯，監視不同電路元件。
- Solution, 它提供了對算法和解的接口。使用它的成員，你可以設定算法參數，解電路，查看解的屬性，比如解所花費的循環次數。

下面將詳細介紹這幾個接口。

TEXT 接口介紹

Text 接口是最簡單的，但是也是最有用的接口。你可以借助它在 COM 接口中執行 OpenDSS 命令。這樣，COM 用戶可以使用任何一個 OpenDSS 命令。

下述是使用 text 接口的例子。

```
' Load in an example circuit
DSSText.Command = "Compile 'C:\example\IEEE123Master.dss'"
' Create a new capacitor
DSSText.Command = "New Capacitor.C1 Bus1=1 Phases=3 kVAR=1200"
DSSText.Command = "~ Enabled=false" ' You can even use ~
' Change the bus for Line L1
DSSText.Command = "Line.L1.Bus1 = 5"
```

如果執行的命令是顯示一個結果，這個結果可以用 Text 的 Result 屬性來獲得：

```
' Export voltage to a csv file
Dim Filename As String
DSSText.Command = "Export Voltages"
```

```
Filename = DSSText.Result
MsgBox("File saved to: " & Filename)
```

電路接口（CIRCUIT INTERFACE）介紹

電路接口（circuit interface）可以被用來編輯電路中不同元件的屬性。幾乎所有 OpenDSS 的元件類（比如 Lines, Loads, Capacitors, CapControls, 等）在電路接口中都有一個子對象。這些子對象還有一些便利的函數可以使用戶在循環中遍歷所有的成員元件。例如，下面的代碼會遍歷所有電路中的負荷（loads）把 kW 增加 20%:

```
' Step through every load and scale it up
Dim iLoads As Integer ' Track what load we're on

iLoads = DSSCircuit.Loads.First
While iLoads
    ' Scale load by 120%
    DSSCircuit.Loads.kW = DSSCircuit.Loads.kW * 1.2
    ' Move to next load
    iLoads = DSSCircuit.Loads.Next
End While
```

如果你想要編輯一個具體的元件，使用 SetActiveElement 方法，ActiveDSSElement 的接口如下:

```
' Set a capacitor's rated kVAR to 1200
DSSCircuit.SetActiveElement("Capacitor.C83")
DSSCircuit.ActiveDSSElement.Properties("kVAR").Val = 1200
```

這種方法和簡單的 text 接口可以取得相同的結果，如下,根據自己的需要選擇方法。

```
' Does the same thing as the previous snippet
DSSText.Command = "Capacitor.C83.kVAR = 1200"
```

電路接口的另外一個有用的特性是可以獲得潮流計算（power flow）的結果，比如電壓，損耗等。下面的例子獲得母線的名字和電壓:

```
' Get bus voltages
Dim BusNames As String()
Dim Voltages As Double()
BusNames = DSSCircuit.AllBusNames
Voltages = DSSCircuit.AllBusVmagPu

' See what an arbitrary bus's voltage is
MsgBox(BusNames(5) & "'s voltage mag in per unit is: " & Voltages(5))
```

更多細節在 OpenDSS manual 的 COM section 的 “All commands” 中 (比如 AllElementLoses, AllNodeVmagByPhase)

算法和解的接口（SOLUTION INTERFACE）介紹

算法和解的接口（Solution Interface）被用來查看和控制 OpenDSS 的計算過程和控制過程。這包括解電路，設定解的模式，查看是否收斂，報告 time 和 duty 模式中的時間段，等等功能。

最基本的功能是,這個接口可以讓我們解電路, 如下:

```
' Solve the Circuit
DSSSolution.Solve()
If DSSSolution.Converged Then
    MsgBox("The Circuit Solved Successfully")
End If
```

你也可以在更多細節上控制解電路。比如, 下面的例子仿真 30 秒電路反應在加入一個大負荷後。

```
' Model effects of a large load pickup 30 seconds into a simulation
DSSText.Command = _
    "New Monitor.Mon1 element=Line.L100 mode=0"
DSSSolution.StepSize = 1      ' Set step size to 1 sec
DSSSolution.Number = 30      ' Solve 30 seconds of the simulation
' Set the solution mode to duty cycle, which forces loads to use their
' "duty cycle" loadshape and allows time based simulation
DSSSolution.Mode = OpenDSSengine.SolveModes.dssDutyCycle
DSSSolution.Solve()

DSSCircuit.Enable("Load.L1")  ' Enable the load
DSSSolution.Number = 30      ' Solve another 30 seconds of simulation
DSSSolution.Solve()

MsgBox("Seconds Elapsed: " & DSSSolution.Seconds)
' Plot the voltage for the 60 seconds of simulation
DSSText.Command = "Plot monitor object=Mon1 Channels=(1,3,5)"
```

電路接口 (circuit interface) 提供許多很具體的解電路背後的方法和資訊接口。使用不同的解命令 (比如, Solve, SolveDirect, SolveNoControl, 等...) 和其他函數如 SystemYChanged 和 MostIterationsDone, 可以獲得解的細節或者在細節上控制解的過程。使用 CheckControls 及其他方法, 連同 DSSCircuit.CtrlQueue 接口, 可以實現用戶定義的控制策略。更多細節可以參閱 OpenDSS TechNotes。

VISUAL BASIC 例子

前面提到的 VB 例子在此處匯總。

```
' *****
' * Initialize OpenDSS
' *****
' Declare the OpenDSS related variables
Dim DSSObj As OpenDSSEngine.DSS
Dim DSSText As OpenDSSEngine.Text
Dim DSSCircuit As OpenDSSEngine.Circuit
Dim DSSSolution As OpenDSSEngine.Solution

' Instantiate the OpenDSS Object
DSSObj = New OpenDSSEngine.DSS

' Start up the Solver
If Not DSSObj.Start(0) Then
    MsgBox("Unable to start the OpenDSS Engine")
    Return
End If

' Set up the Text, Circuit, and Solution Interfaces
DSSText = DSSObj.Text
DSSCircuit = DSSObj.ActiveCircuit
DSSSolution = DSSCircuit.Solution

' *****
' * Examples Using the DSSText Object
' *****
' Load in an example circuit
DSSText.Command = "Compile 'C:\example\IEEE123Master.dss'"
' Create a new capacitor
DSSText.Command = "New Capacitor.C1 Bus1=1 Phases=3 kVAR=1200"
DSSText.Command = "~ Enabled=false" ' You can even use ~
' Change the bus for Line L1
DSSText.Command = "Line.L1.Bus1 = 5"

' Export voltage to a csv file
Dim Filename As String
DSSText.Command = "Export Voltages"
Filename = DSSText.Result
MsgBox("File saved to: " & Filename)

' *****
' * Examples Using the DSSCircuit Object
' *****
' Step through every load and scale it up
Dim iLoads As Integer ' Track what load we're on

iLoads = DSSCircuit.Loads.First
While iLoads
    ' Scale load by 120%
    DSSCircuit.Loads.kW = DSSCircuit.Loads.kW * 1.2
    ' Move to next load
    iLoads = DSSCircuit.Loads.Next
End While

' Set a capacitor's rated kVAR to 1200
DSSCircuit.SetActiveElement("Capacitor.C83")
DSSCircuit.ActiveDSElement.Properties("kVAR").Val = 1200
' Get bus voltages
```

```
Dim BusNames As String()
Dim Voltages As Double()
BusNames = DSSCircuit.AllBusNames
Voltages = DSSCircuit.AllBusVmagPu

' See what an arbitrary bus's voltage is
MsgBox(BusNames(5) & "'s voltage mag in per unit is: " & Voltages(5))

' *****
' * Examples Using the DSSSolution Object
' *****
' Solve the Circuit
DSSSolution.Solve()
If DSSSolution.Converged Then
    MsgBox("The Circuit Solved Successfully")
End If

' Model effects of a large load pickup 30 seconds into a simulation
DSSText.Command = _
    "New Monitor.Mon1 element=Line.L100 mode=0"
DSSSolution.StepSize = 1          ' Set step size to 1 sec
DSSSolution.Number = 30           ' Solve 30 seconds of the simulation
' Set the solution mode to duty cycle, which forces loads to use their
' "duty cycle" loadshape and allows time based simulation
DSSSolution.Mode = OpenDSSengine.SolveModes.dssDutyCycle
DSSSolution.Solve()

DSSCircuit.Enable("Load.L1")      ' Enable the load
DSSSolution.Number = 30           ' Solve another 30 seconds of simulation
DSSSolution.Solve()

MsgBox("Seconds Elapsed: " & DSSSolution.Seconds)
' Plot the voltage for the 60 seconds of simulation
DSSText.Command = "Plot monitor object=Mon1 Channels=(1,3,5)"
```

MATLAB 例子

以下是 Matlab 的 COM 接口的例子

```
clc
clear all
close all

% *****
% * Initialize OpenDSS
% *****
% Instantiate the OpenDSS Object
DSSObj = actxserver('OpenDSSEngine.DSS');

% Start up the Solver
if ~DSSObj.Start(0),
    disp('Unable to start the OpenDSS Engine')
    return
end

% Set up the Text, Circuit, and Solution Interfaces
DSSText = DSSObj.Text;
DSSCircuit = DSSObj.ActiveCircuit;
DSSSolution = DSSCircuit.Solution;

% *****
% * Examples Using the DSSText Object
% *****
% Load in an example circuit
DSSText.Command = 'Compile "C:\example\IEEE123Master.dss"';
% Create a new capacitor
DSSText.Command = 'New Capacitor.C1 Bus1=1 Phases=3 kVAR=1200';
DSSText.Command = '~ Enabled=false'; % You can even use ~
% Change the bus for Line L1
DSSText.Command = 'Line.L1.Bus1 = 5';

% Export voltage to a csv file
DSSText.Command = 'Export Voltages';
Filename = DSSText.Result;
disp(['File saved to: ' Filename])

% *****
% * Examples Using the DSSCircuit Object
% *****
% Step through every load and scale it up

iLoads = DSSCircuit.Loads.First;
while iLoads,
    % Scale load by 120%
    DSSCircuit.Loads.kW = DSSCircuit.Loads.kW * 1.2;
    % Move to next load
    iLoads = DSSCircuit.Loads.Next;
end
```

```
% Set a capacitor's rated kVAR to 1200
DSSCircuit.SetActiveElement('Capacitor.C83');
DSSCircuit.ActiveDSSElement.Properties('kVAR').val = '1200';

% Get bus voltages
BusNames = DSSCircuit.AllBusNames;
Voltages = DSSCircuit.AllBusVmagPu;

% See what an arbitrary bus's voltage is
disp([BusNames{5} ' 's voltage mag in per unit is: '
num2str(Voltages(5))])

% *****
% * Examples Using the DSSSolution Object
% *****

% Solve the Circuit
DSSSolution.Solve();
if DSSSolution.Converged,
    disp('The Circuit Solved Successfully')
end

% Model effects of a large load pickup 30 seconds into a simulation
DSSText.Command = ...
    'New Monitor.Mon1 element=Line.L100 mode=0';
DSSSolution.StepSize = 1;          % Set step size to 1 sec
DSSSolution.Number = 30;           % Solve 30 seconds of the simulation
% Set the solution mode to duty cycle, which forces loads to use their
% 'duty cycle' loadshape and allows time based simulation
DSSSolution.Mode = 6;              % Code for duty cycle mode
DSSSolution.Solve();

DSSCircuit.Enable('Load.L1');      % Enable the load
DSSSolution.Number = 30;           % Solve another 30 seconds of simulation
DSSSolution.Solve();

disp(['Seconds Elapsed: ' num2str(DSSSolution.Seconds)])
% Plot the voltage for the 60 seconds of simulation
DSSText.Command = 'Plot monitor object=Mon1 Channels=(1,3,5)';
```


PYTHON 例子

```

import win32com.client

# *****
# * Initialize OpenDSS
# *****
# Instantiate the OpenDSS Object
try:
    DSSObj = win32com.client.Dispatch("OpenDSSEngine.DSS")
except:
    print "Unable to start the OpenDSS Engine"
    raise SystemExit

# Set up the Text, Circuit, and Solution Interfaces
DSSText = DSSObj.Text
DSSCircuit = DSSObj.ActiveCircuit
DSSSolution = DSSCircuit.Solution

# *****
# * Examples Using the DSSText Object
# *****
# Load in an example circuit
DSSText.Command = r"Compile 'C:\example\IEEE123Master.dss'"
# Create a new capacitor
DSSText.Command = "New Capacitor.C1 Bus1=1 Phases=3 kVAR=1200"
DSSText.Command = "~ Enabled=false" # You can even use ~
# Change the bus for Line L1
DSSText.Command = "Line.L1.Bus1 = 5"

# Export voltage to a csv file
DSSText.Command = "Export Voltages"
Filename = DSSText.Result
print "File saved to: " + Filename

# *****
# * Examples Using the DSSCircuit Object
# *****
# Step through every load and scale it up
iLoads = DSSCircuit.Loads.First
while iLoads:
    # Scale load by 120%
    DSSCircuit.Loads.kW = DSSCircuit.Loads.kW * 1.2
    # Move to next load
    iLoads = DSSCircuit.Loads.Next

# Set a capacitor's rated kVAR to 1200
DSSCircuit.SetActiveElement("Capacitor.C83")
DSSCircuit.ActiveDSElement.Properties("kVAR").Val = 1200

# Get bus voltages
BusNames = DSSCircuit.AllBusNames
Voltages = DSSCircuit.AllBusVmagPu

```

```
# See what an arbitrary bus's voltage is
print BusNames[5] + "'s voltage mag in per unit is: " + \
      str(Voltages[5])

# *****
# * Examples Using the DSSSolution Object
# *****
# Solve the Circuit
DSSSolution.Solve()
if DSSSolution.Converged:
    print "The Circuit Solved Successfully"

# Model effects of a large load pickup 30 seconds into a simulation
DSSText.Command = \
    "New Monitor.Mon1 element=Line.L100 mode=0"
DSSSolution.StepSize = 1          # Set step size to 1 sec
DSSSolution.Number = 30          # Solve 30 seconds of the simulation
# Set the solution mode to duty cycle, which forces loads to use their
#   "duty cycle" loadshape and allows time based simulation
DSSSolution.Mode = 6             # Code for duty cycle mode
DSSSolution.Solve()

DSSCircuit.Enable("Load.L1")      # Enable the load
DSSSolution.Number = 30          # Solve another 30 seconds of simulation
DSSSolution.Solve()

print "Seconds Elapsed: " + str(DSSSolution.Seconds)
# Plot the voltage for the 60 seconds of simulation
DSSText.Command = "Plot monitor object=Mon1 Channels=(1,3,5) "
```

其他資源

在你探索 OpenDSS 廣泛的應用時，你會遇到本簡介沒有介紹的情況。在這時，你可以參閱 OpenDSS 開發者和用戶社區提供的以下資源。

下一步該去哪兒？

通過這個入門簡介，你應該可以開始使用 OpenDSS。更多的有關 OpenDSS 特性的例子可以在安裝目錄下找到，包括：


- 更多的 OpenDSS 代碼範例，在 “Examples/Scripts” 目錄中。
- 更多的例子有關於：解的模式，比如時間仿真, duty-cycle 仿真, 年能量仿真（annual energy simulations），Monte Carlo analysis, 動態仿真（dynamic analysis），諧波仿真（harmonic analysis），故障仿真（fault studies），電容器位置優化（capacitor placement optimization），地磁感應電流（geomagnetically induced current studies），等
- 關於電路元件的例子, 如
 - 調壓器和及其控制，電容器及其控制，集中控制的 volt-VAR 優化控制
 - 各種發電機模型，如傳統的 PV（功率電壓）發電機，固定功率因數發電機，逆變器，太陽能發電機，感應發電機，儲能電源等。
 - 斷路器等（Fuses, reclosers, relays, and faults）
 - 負荷曲線（load shapes），增長曲線（growth curves），諧波頻譜（harmonic spectrums），線路幾何信息（line geometries），溫度曲線（temperature curves），能量計量器（energy meters），和監視器（monitors）
- 範例包含以下幾種語言的 COM 接口
 - MATLAB
 - Python
 - Visual Basic for Applications through Excel
- 在 EPRI TestCircuits 目錄中，有一系列真實的配網模型
- 一個簡單的輸電網模型在 /Stevenson 目錄中
- 更多的 OpenDSS 代碼範例，在 “Examples/Scripts” 目錄中。
- 更多的例子有關於：解的模式，比如時間仿真, duty-cycle 仿真, 年能量仿真（annual energy simulations），Monte Carlo analysis, 動態仿真（dynamic analysis），諧波仿真（harmonic analysis），故障仿真（fault studies），電容器位置優化（capacitor placement optimization），地磁感應電流（geomagnetically induced current studies），等
- 關於電路元件的例子, 如
 - 調壓器和及其控制，電容器及其控制，集中控制的 volt-VAR 優化控制
 - 各種發電機模型，如傳統的 PV（功率電壓）發電機，固定功率因數發電機，逆變器，太陽能發電機，感應發電機，儲能電源等。
 - 斷路器等（Fuses, reclosers, relays, and faults）
 - 負荷曲線（load shapes），增長曲線（growth curves），諧波頻譜（harmonic spectrums），線路幾何信息（line geometries），溫度曲線

- (temperature curves)，能量計量器 (energy meters)，和監視器 (monitors)
- 範例包含以下幾種語言的 COM 接口
 - MATLAB
 - Python
 - Visual Basic for Applications through Excel
 - 在 EPRI TestCircuits 目錄中，有一系列真實的配網模型
 - 一個簡單的輸電網模型在/Stevenson 目錄中

其他的介紹和培訓文件在 OpenDSS 的安裝目錄中可找到，在 SourceForge 的培訓目錄中也可找到, 點擊這裏 [here](#).

參考資源

OpenDSS 其他有關編程和 COM 接口的參考資料包括：

- **In-program help interface**, 程序內幫助，可以通過 Help > DSS Help 或者  按鈕打開。簡明介紹特性和命令。
- **OpenDSS manual**, 在安裝路徑目錄中。詳細介紹特性，COM 接口等。
- **OpenDSS TechNotes**, 在安裝路徑目錄中。一系列介紹 COM 接口，OpenDSS 特性，及其他具體應用（如變壓器模型，太陽能發電機）的文章。
- **OpenDSS wiki**, 點擊 [here](#) 它尤其提供 COM 接口的有關資源。
- **Type library browser (TLB)** - 例如 Microsoft Office' s Visual Basic Editor - 可以用來查看 COM 對象。打開 Microsoft Office' s Visual Basic Editor，在 reference (在 Tools 菜單中) 中填加 OpenDSSEngine. 然後你可以使用"Object Browser"（在 View 菜單中）瀏覽 OpenDSSEngine 的庫。

更多幫助

如果你需要更多幫助，在 SourceForge 網站的 OpenDSS forums 中，OpenDSS 用戶群會為你提供更多解答。論壇點擊 [here](#).