Introduction to PowerWorld Simulator: Interface and Common Tools

I15: SimAuto Overview

SimAuto is a COM automation server that allows Simulator to be controlled from an external application

2001 South First Street
Champaign, Illinois 61820
+1 (217) 384.6330

support@powerworld.com
http://www.powerworld.com

Possible Applications

- Use Visual Basic for Applications to load generator cost data and unit commitment from Access or SQL, solve the OPF, and write solutions back to the database
- Use MATLAB to calculate generator parameters over time and solve a sequence of instantaneous power flows in Simulator

Using SimAuto

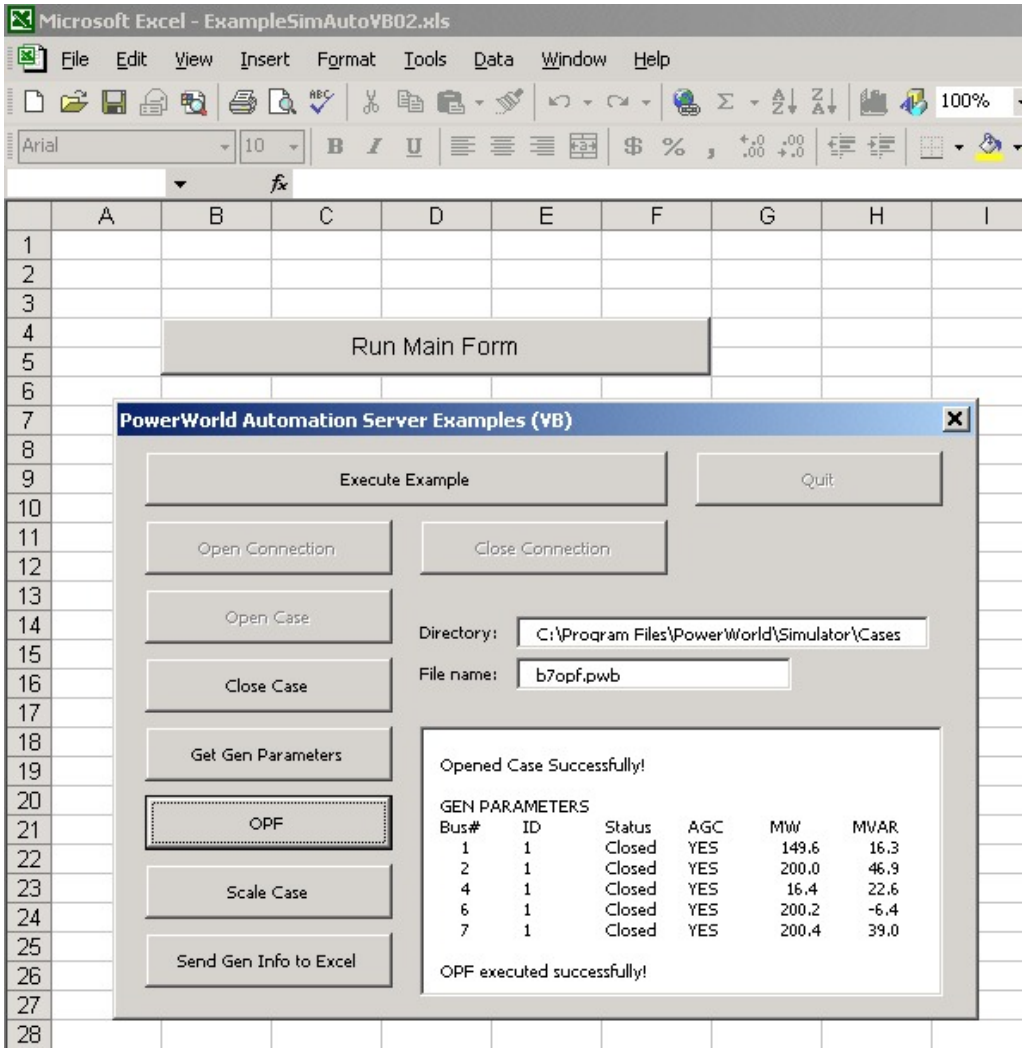
- Reference the Simulator Type Library in your programming environment
- Connect to the Automation Server
- Interface with Simulator using SimAuto Functions

ChangeParameters
ChangeParametersSingleElement
ChangeParametersMultipleElement
ChangeParametersMultipleElementFlatInput
CloseCase
GetCaseHeader
GetParametersSingleElement
GetParametersMultipleElement
GetParametersMultipleElementFlatOutput
ListOfDevices
ListOfDevicesAsVariantStrings
ListOfDevicesAsVariantStrings
ListOfDevicesFlatOutput
OpenCase
ProcessAuxFile
RunScriptCommand
GetFieldList
SaveState
LoadState
SaveCase
SendToExcel
WriteAuxFile

Visual Basic OPF Demonstration

- Excel VBA application allows user to perform several operations in any order
  - Solve OPF
  - Scale case
  - Write generator records to Excel
- Open Excel file ExampleSimAutoVB02.xls and select **Enable Macros** if prompted (code will not function if Excel settings do not allow the use of macros)
- Open Simulator
- Click **Run Main Form** button, then **Open Connection**, then **Open Case** to activate the other options

Visual Basic OPF Demonstration



SimAuto Tips

- This example and others may be downloaded from the PowerWorld website
- Code for examples may be accessed in Excel by selecting the Design Mode and Visual Basic Editor from the Visual Basic toolbar
- Simulator commands are identical to those used in SCRIPT language

SimAuto Tips

- Simulator objects and data fields are accessed as they are in DATA sections of script files
  - Reference object types with identical syntax

SimAuto Functions

- ChangeParametersSingleElement(ObjectType, ParamList, Values)
  - ObjectType : String
    - The type of object for which parameters are being changed, e.g. “BUS”.
  - ParamList : Variant
    - A variant array storing strings that are Simulator object field variables, e.g. “BusNum”.
    - Must contain the key fields for the objecttype.
  - Values : Variant
    - A variant array storing variants (integer, string, single, etc.) that are the values for each of the fields in the ParamList.
  - Output
    - Returns any errors in the first element, i.e. Output(0)

SimAuto Functions

- ChangeParametersMultipleElement(ObjectType, ParamList, ValueList)
  - ObjectType : String
    - The type of object for which parameters are being changed, e.g. “BUS”.
  - ParamList : Variant
    - A variant array storing strings that are Simulator object field variables, e.g. “BusNum”.
    - Must contain the key fields for the objecttype.
  - ValueList : Variant
    - A variant array storing arrays of variants.
    - Create variant arrays (one for each element being changed) with values corresponding to the fields in ParamList. Insert each of these variant arrays into ValueList.
  - Output
    - Returns any errors in the first element, i.e. Output(0)

SimAuto Functions

- ChangeParametersMultipleElement – Sample VBA Code

```
Dim ValueList(1), ParamList as Variant
Dim Output as Variant

ParamList = Array("BusNum", "AreaName")
ValueList(0) = Array(1,"Right")
ValueList(1) = Array(2,"Left")
Output = SimAuto.ChangeParametersMultipleElement("BUS",
                                        ParamList,ValueList)
```

SimAuto Functions

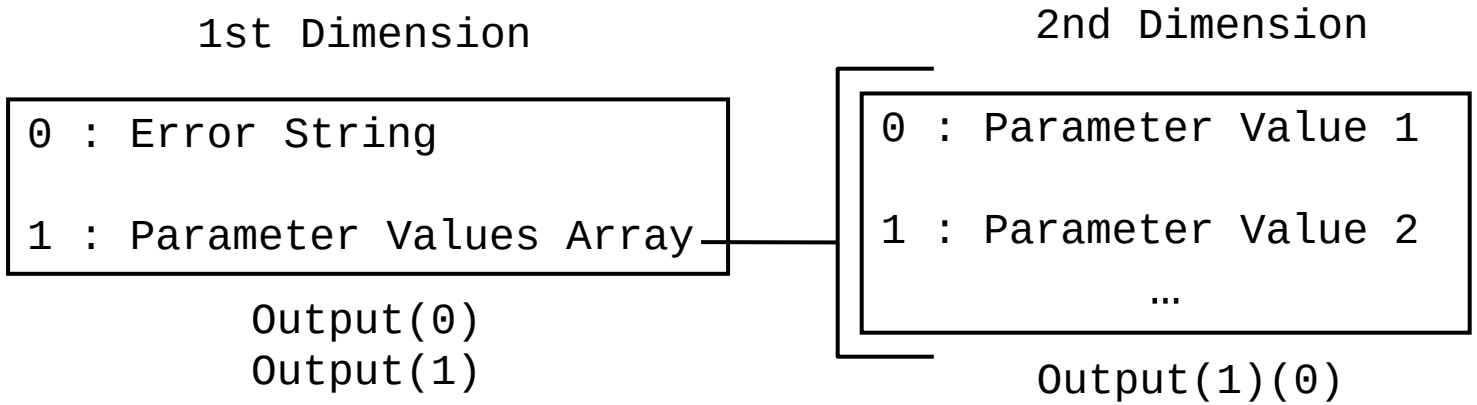
- ChangeParametersMultipleElementFlatInput(ObjectType,ParamList, NoOfObjects,ValueList)
  - ObjectType : String
    - Type of object for which parameters are being changed
  - ParamList : Variant
    - A variant array storing strings that are Simulator object field variables, e.g. “BusNum”.
    - Must contain the key fields for the object type.
  - NoOfObjects : Integer
    - Number of devices for which values are being passed
  - ValueList : Variant
    - Single-dimensional variant array storing a list of variants (integer, single, string, etc.) representing the values corresponding to ParamList for all devices being changed
    - All parameters for the first object are listed first followed by all of the parameters for the second object, etc.
      - ValueList = Array(Obj1Param1, Obj1Param2, …Obj1ParamM, Obj2Param1, Obj2Param2, … Obj2ParamM, Obj3Param1, …ObjNParam1, …ObjNParamM)
  - Output
    - Returns any errors in the first element, i.e. Output(0)

SimAuto Functions

- GetParametersSingleElement(ObjectType, ParamList, Values)
  - ObjectType : String
    - The type of object for which parameters are being retrieved, e.g. “BUS”.
  - ParamList : Variant
    - A variant array storing strings that are Simulator object field variables, e.g. “BusNum”.
    - Must contain the key fields for the object type.
  - Values : Variant
    - A variant array storing variants (integer, string, single, etc.) that are the values for each of the fields in the ParamList.
    - Values must be passed in for the key fields
    - Values other than key fields should be set to zero

SimAuto Functions

- GetParametersSingleElement Output
  - First element contains any errors
  - Second element is a one dimensional array containing values corresponding to fields specified in ParamList



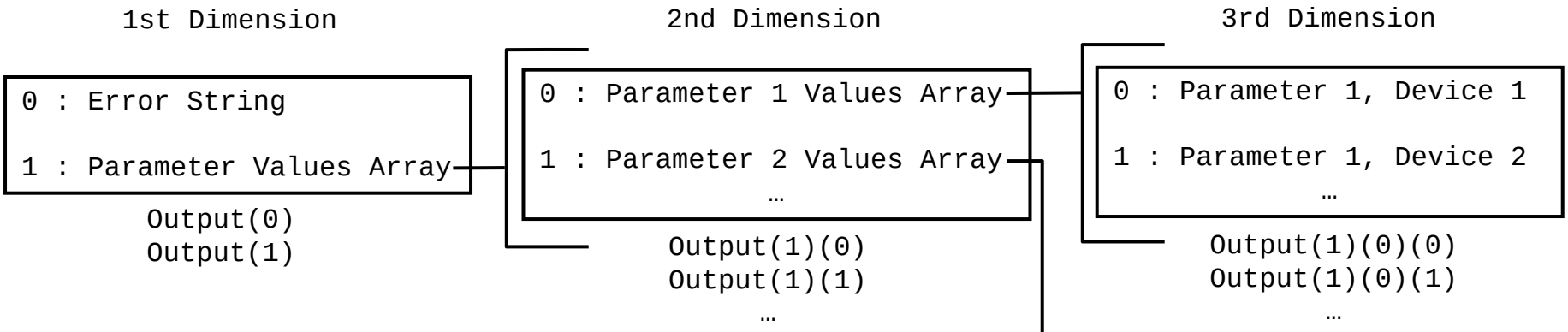

```
Example
Use B7Flat.pwb to get generator MW at bus 1
Output=GetParametersSingleElement
  (“BUS”,Array(“BusNum”,”BusGenMW”),Array(1,0))
Output(0) => error message string
Output(1)(0) => 1 => bus number for bus 1
Output(1)(1) => 101.85 => generator MW for bus 1
```

SimAuto Functions

- GetParametersMultipleElement(ObjectType, ParamList, FilterName)
  - ObjectType : String
    - The type of object for which parameters are being retrieved, e.g. “BUS”.
  - ParamList : Variant
    - A variant array storing strings that are Simulator object field variables, e.g. “BusNum”.
    - Must contain the key fields for the object type.
  - FilterName : String
    - Name of a pre-defined advanced filter that will limit the objects returned.
    - Pass an empty string to return all objects of the specified type.
  - Output
    - Set of nested arrays containing the parameter values for the device type requested
    - Number of arrays returned depends on the number of fields in ParamList

SimAuto Functions

- GetParametersMultipleElement Output



Example
Use B7Flat.pwb to get generator MW at each bus
Output=GetParametersMultipleElement
(“BUS”,Array(“BusNum”,”BusGenMW”),””)
Output(0) => error message string
Output(1)(0) => array of bus numbers
Output(1)(1) => array of generator MW values
Output(1)(0)(0) => 1 => bus number for the first bus
Output(1)(0)(1) => 2 => bus number for the second bus
Output(1)(1)(0) => 101.85 => gen MW for the first bus
Output(1)(1)(1) => 170.08 => gen MW for the second bus

SimAuto Functions

- GetParametersMultipleElementFlatOutput (ObjectType,ParamList,FilterName)
  - Inputs are handled in the same manner as GetParametersMultipleElement
  - Output
    - Single-dimensional array instead of nested arrays
    - Array(errorstring, NumberOfObjectsReturned, NumberOfFieldsPerObject, Ob1Fld1, Ob1Fld2, …, Ob(n)Fld(m-1), Ob(n)Fld(m))

SimAuto Functions

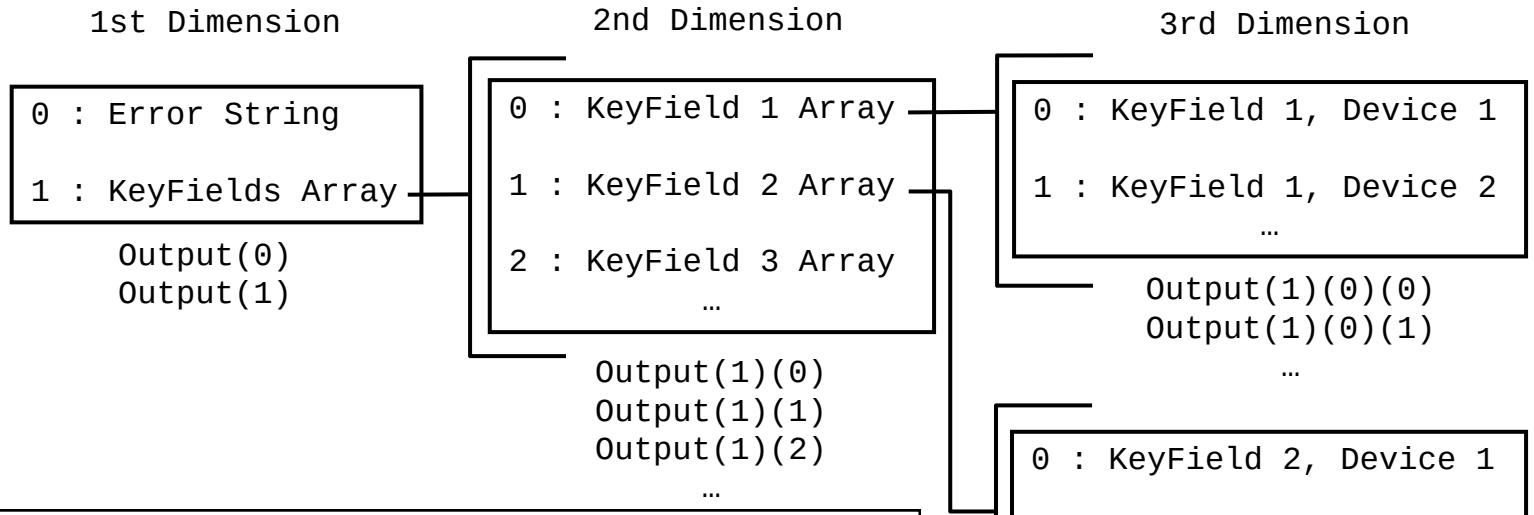
- GetFieldList(ObjectType)
  - Returns all fields associated with a given object type.
  - ObjectType : String
    - Type of object for which fields are requested, e.g. “BUS”.
  - Output
    - First element is the error string
    - Second element is an n x 4 array of fields
      - Similar to information obtained from Export Case Object Fields…
      - (n,0) specifies the key and required fields
        - » Key - *1*,*2*, etc.
        - » Secondary Key – *A*, *B*, etc.
        - » Required – **
      - (n,1) variablename of the field
      - (n,2) type of data stored in the field (integer, string, real)
      - (n,3) field description

SimAuto Functions

- ListOfDevices(ObjectType,FilterName)
  - ObjectType : String
    - Type of object for which devices are being acquired.
  - FilterName : String
    - Name of a pre-defined advanced filter that will limit the objects returned.
    - Pass an empty string to return all objects of the specified type.
  - Output
    - Set of nested arrays containing the key field values for the type of object requested.
    - Number of arrays returned depends on the object type selected.
    - Values in the arrays are strongly typed, i.e. bus numbers are returned as long integers instead of as a variant
      - Use ListOfDevicesAsVariantStrings to return values as variants

SimAuto Functions

- ListOfDevices Output



Example
Use B7Flat.pwb to get list of generators
Output = ListOfDevices("GEN","")
Output(0) => error message string
Output(1)(0) => array of bus numbers
Output(1)(1) => array of generator IDs
Output(1)(0)(0) => 1 => bus number for the first gen
Output(1)(0)(1) => 2 => bus number for the second gen
Output(1)(1)(0) => "1" => gen ID for the first gen
Output(1)(1)(1) => "1" => gen ID for the second gen

SimAuto Functions

- ListOfDevicesFlatOutput(ObjectType, FilterName)
  - Inputs same as ListOfDevices
  - Output
    - Single-dimensional array of variants
    - Array(errorString, NumberOfObjectsReturned, NumberOfFieldsPerObject, Ob1Fld1, Ob1Fld2,…, Ob(n)Fld(m-1), Ob(n), Fld(m))