IOWA STATE UNIVERSITY

Department of Economics, Department of Electrical & Computer Engineering

Report on:

ERCOT PNNL Contract 401882: *Start Date* 3/19/2018

Development of an Integrated Transmission and Distribution Test System to Evaluate Transactive Energy Systems

ISU Project Team:

PI Leigh Tesfatsion & Co-PI Zhaoyu Wang

Grad Research Assistant: Swathi Battula

PNNL/ISU Web Conference, 22 March 2019

ERCOT Contract: Presentation Outline

- ❑ Original Task/Milestone Schedule: M1-M3

- ❑ Updated Task/Milestone Schedule: M1, M2, M3.1, M3.2, M3.3

- ❑ M3.2: Completed Work on AMES V5.0

- ❑ M3.2: Work in Progress on AMES V5.0 and ERCOT 200-Bus Test Case

  - ✓ ***Key Milestone:*** ERCOT 200-Bus Test Case now runs for multiple successive simulated days with no compilation or run-time errors

- ❑ Questions for PNNL regarding modeling of GLD's house object – file name `house_e.cpp'

- ❑ Preparation for AMES V5.0 J-Unit Testing: Event Sequencing

Original Task & Milestone Schedule

Milestone	Date Due	Original Description
M1	May 31, 2018	5-zone model of the old ERCOT system, posted to a web repository.
M2	Sep 30, 2018	Nodal model of the new ERCOT system, posted to a web repository.
M3	Sep 30, 2018	Submitted conference or journal paper on this work.

Updated Task & Milestone Schedule

Milestone	Date Due	Date Delivered	Fuller Descriptions of Actual Work
M1* DONE	May 31, 2018	June 5, 2018	Development of 8-Bus ERCOT model (with nodal locational marginal pricing); grid/load/gen data posted at PNNL repository
M2.1 DONE	Sept 30, 2018	August 1, 2018	Basic 8-Bus ERCOT Test System, implemented via AMES V3.1, posted at https://github.com/ITDProject/ERCOTTestSystem
M2.2 DONE	Sept 30, 2018	August 24, 2018	8-Bus ERCOT Test System (with wind power), implemented via AMES V3.2, posted at https://github.com/ITDProject/ERCOTTestSystem
M3.1 DONE	Sept 30, 2018	August 31, 2018	200-Bus ERCOT Test System (with wind power), implemented via AMES V3.2, posted at https://github.com/ITDProject/ERCOTTestSystem/tree/master/ERCOT_Test_Systems/The_200Bus_ERCOT_Test_System
M3.2**	July 31, 2019		200-Bus ERCOT Test System (with Non-Dispatchable Generation), implemented via AMES V5.0, to be posted at PNNL/ISU repositories.
M3.3**	July 31, 2019		Paper to be submitted that focuses on the development of the ERCOT Test Systems

- *** M1 Modification (Ok'd by PNNL):** For M1 we have skipped the modeling of the old (zonal) ERCOT system and instead directly worked to develop an 8-bus model of the new (nodal) ERCOT system.
- **** M3 Modification: Contract extension through July 31, 2019 received from PNNL on March 4, 2019, for completion of task M3**

Summary of AMES V5.0 Work to Date for M3.2

- **Extension of AMES V5.0 Capabilities for Milestone M3.2**
  - [DONE] Coding for Daily DAM SCUC optimization
  - [DONE] Coding for RTM SCED optimization every M minutes with a user-specified M
  - [DONE] Coding for FNCS integration to enable network co-simulation
  - [DONE] Detailed documentation for analytical DAM SCUC/SCED optimization in AMES V5.0
  - [DONE] Basic documentation for AMES V5.0, including a detailed list for all parameters/flags and initial state variables that need user configuration.

Summary of AMES V5.0 Work to Date for M3.2 … Continued

- **[DONE]**

— **Modified `PSST’ Code**

- To ensure correct refreshing of initial DAM/RTM conditions for multiple-day runs.
- To report DAM LMPs and GenCo Commitments back to the user.
- To read ‘startup’ and ‘shutdown’ cost components from AMES
- To produce output messages related to solver, e.g. status of the solver, termination condition of the solver
- To include the parameter ‘Maximum Time Limit’ – to allow the solver to terminate after the prescribed time has elapsed

— **Verification Tests Done**

- Verified ‘DAM SCUC’ outcomes for their correctness for simple test cases
- Verified ‘RTM SCED’ outcomes for their correctness for simple test cases with RTM running every five minutes (i.e., M=5)
- Verified that AMES V5.0 runs for multiple days
- Verified that all the cost components from AMES are read correctly into the SCUC formulation

Summary of AMES V5.0 Work to Date for M3.2…Continued

- ❑ VerTestCaseBaseCase **DONE**
  - This test case produces SCUC/SCED outcomes under the following conditions:
    - ✓ Transmission congestion is absent
    - ✓ Minimum power generation limits are taken to be zero
    - ✓ Start up, shut down and no-load costs are taken to be zero
    - ✓ Minimum up-time and down-time values are taken to be 0 (hr)
    - ✓ No ramping limits
    - ✓ Day-ahead and real-time load forecasts are set equal
  - This test case provides a base case for later comparison purposes.
- ❑ VerTestCaseGenMinPowerLevel **DONE**
  - This test case verifies a generator’s minimum power level is maintained when it is committed, given the above-stated conditions (i) and (iii)-(vi).
- ❑ VerTestCaseUpTimeDownTime **DONE**
  - This test case verifies a generator’s minimum up time and down time are maintained when it is committed, given the above-stated conditions (i) and (iii)-(vi).
- ❑ VerTestCaseMultiDayRun **DONE**
  - This test case verifies DAM/RTM initial conditions are refreshed appropriately when AMES V5.0 is run for multiple successive days.

Note: Files for the above test cases are uploaded at https://github.com/ITDProject/ERCOTTestSystem/tree/dev-source-code/AMES-V5.0/DATA/VerificationTestCases

Summary of Additional Completed and Ongoing AMES V5.0 Verification Test Cases

- VerTestCaseCostComponents

  The purpose of this test case is to verify cost component aspects of the SCUC formulation under the above-stated conditions (i)-(ii) and (iv)-(vi) – i.e., to verify that the SCUC formulation correctly includes no load, start-up, dispatch, and shut-down cost components.

✓ **DONE**

- VerTestCaseCostComponentsNoLoad

  This test case verifies that no load cost is appropriately taken into account in SCUC/SCED formulation given the above-stated conditions (i), (ii) and (iv)-(vi).

✓ To be completed

- VerTestCaseCostComponentsStartUP
- VerTestCaseCostComponentsShutDown

Latest Work on AMES V5.0 … The 200-Bus ERCOT Test Case

- This past week, run-time errors were observed for reading of SCED data while running the 200-bus test case for multiple days.
- Modified AMES V5.0 code to ensure correct reading of SCED data.
- With the latest modifications, AMES V5.0 is now able to run test case ‘AMES_ercot_200bus.dat’ without run-time errors for multiple days.
- Uploaded the modified ‘AMES_ercot_200bus.dat’ file at https://github.com/ITDProject/ERCOTTestSystem/tree/dev-source-code/AMES-V5.0/DATA/ERCOT
- Uploaded the latest code at https://github.com/ITDProject/ERCOTTestSystem/tree/dev-source-code/AMES-V5.0

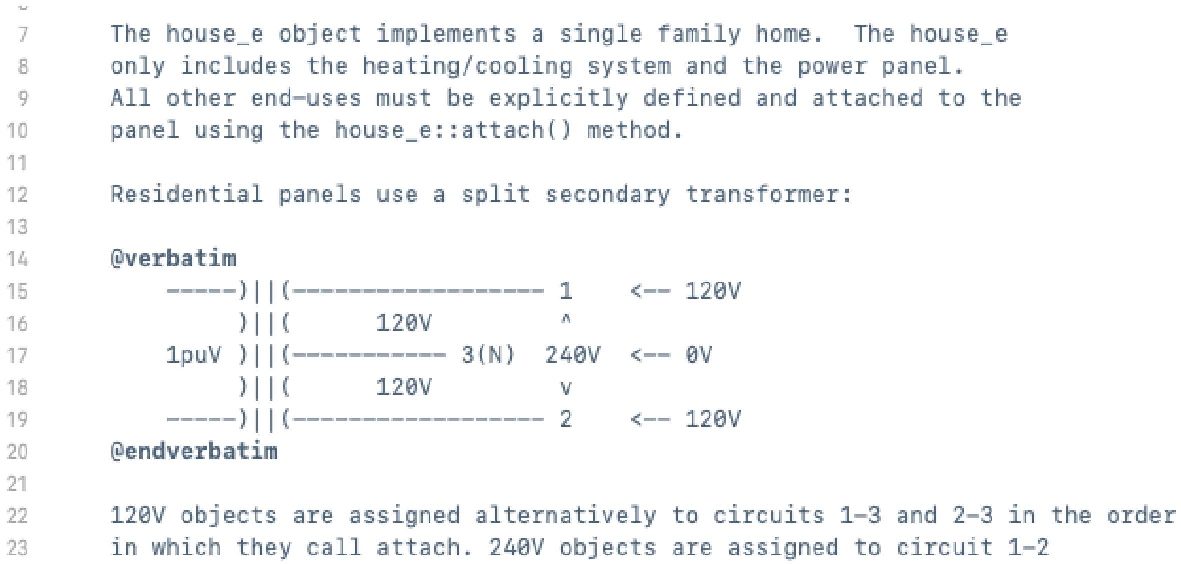
Questions on GLD’s House Object … Preliminaries

- **Query 1:** Does the `house_e.cpp’ appropriately model the voltage being taken into house object from triplex meter object?

Preliminary Materials:

The following code snippet taken from `house_e.cpp’ file indicates that a house can be connected between any of the ‘1-3’ (1N) or ‘2-3’(2N) 120 V circuit of the secondary transformer.

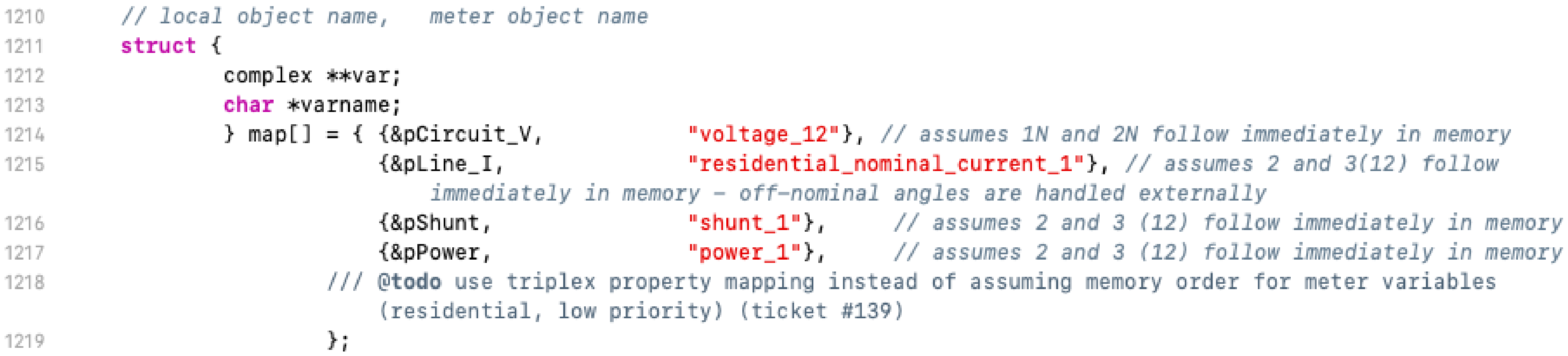
(Note: A triplex meter is used to connect a house to power system and the triplex_meter provides voltage to the house panel.)



```
The house_e object implements a single family home.  The house_e
only includes the heating/cooling system and the power panel.
All other end-uses must be explicitly defined and attached to the
panel using the house_e::attach() method.

Residential panels use a split secondary transformer:

@verbatim
    -----)||(------------------ 1    <-- 120V
         )||(      120V         ^
    1puV )||(----------- 3(N)  240V  <-- 0V
         )||(      120V         v
    -----)||(------------------ 2    <-- 120V
@endverbatim

120V objects are assigned alternatively to circuits 1-3 and 2-3 in the order
in which they call attach. 240V objects are assigned to circuit 1-2
```

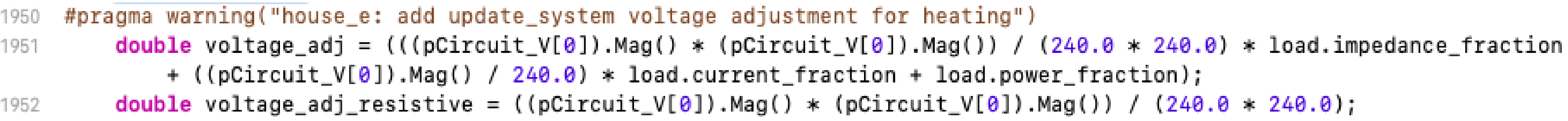
Questions on GLD’s House Object … Preliminaries Continued

The following code snippet (taken from `house_e.cpp’ file) shows that pCircuit_V stores the values of voltage1N and voltage 2N in consecutive memories



```
    // local object name,   meter object name
    struct {
            complex **var;
            char *varname;
            } map[] = { {&pCircuit_V,           "voltage_12"}, // assumes 1N and 2N follow immediately in memory
                        {&pLine_I,              "residential_nominal_current_1"}, // assumes 2 and 3(12) follow immediately in memory - off-nominal angles are handled externally
                        {&pShunt,               "shunt_1"},     // assumes 2 and 3 (12) follow immediately in memory
                        {&pPower,               "power_1"},     // assumes 2 and 3 (12) follow immediately in memory
                    /// @todo use triplex property mapping instead of assuming memory order for meter variables (residential, low priority) (ticket #139)
                    };
```

The following code snippet (taken from `house_e.cpp’ file) shows the magnitude of pCircuit_V[0] taken for calculations.



```
#pragma warning("house_e: add update_system voltage adjustment for heating")
    double voltage_adj = (((pCircuit_V[0]).Mag() * (pCircuit_V[0]).Mag()) / (240.0 * 240.0) * load.impedance_fraction + ((pCircuit_V[0]).Mag() / 240.0) * load.current_fraction + load.power_fraction);
    double voltage_adj_resistive = ((pCircuit_V[0]).Mag() * (pCircuit_V[0]).Mag()) / (240.0 * 240.0);
```

Questions on GLD’s House Object … Continued

Query 1.1: What does pCircuit_V[0] represent? Does it represent voltage1N?

Query 1.2: If pCircuit_V[0] represents voltage1N, then voltage 1N appears to be chosen irrespective of where the house is connected (1N or 2N).

- ✓ Is this right?

Query 1.3: Also, if pCircuit_V[0] represents voltage1N, then -- for a house that is connected between `2-3’ (2N) -- it appears that the `house_e.cpp’ code does not capture the voltage magnitude appropriately.

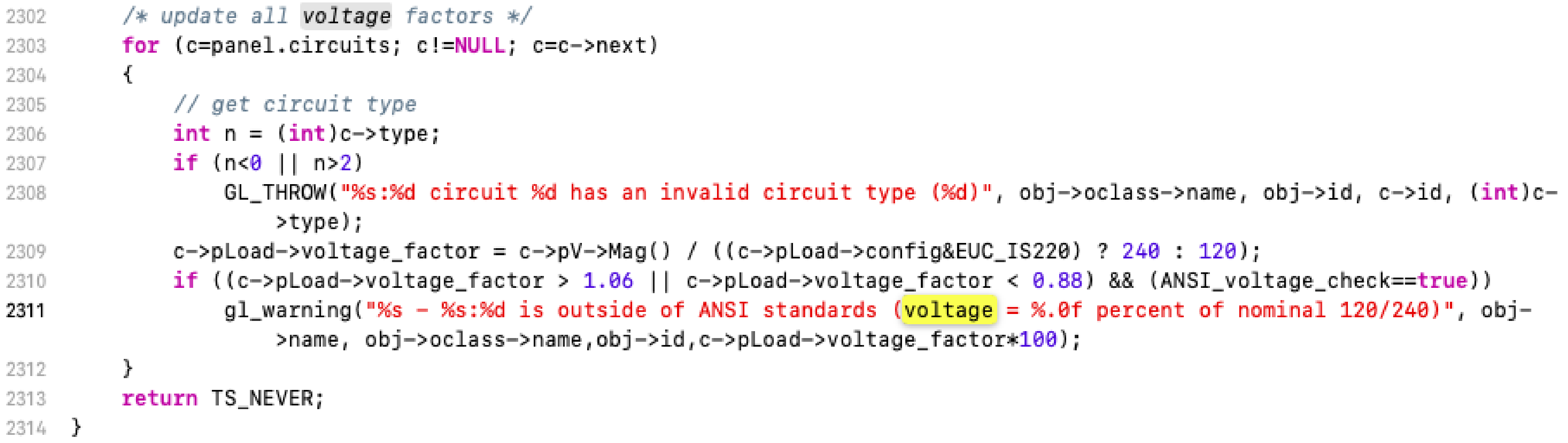
- ✓ Is this right?

Note: Line 1218 in the code says “TODO: use triplex property mapping instead.”

Questions on GLD’s House Object … Continued

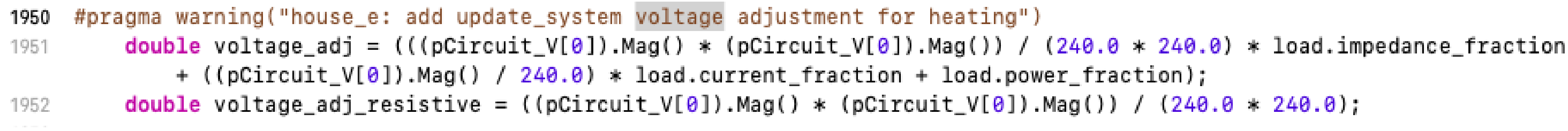
Query 2: Does the `house_e.cpp’ appropriately take into account the nominal voltage (or voltage reference)?

The following code snippet taken from `house_e.cpp’ file indicates that either 120 V or 240 V taken as the nominal voltage.



```
    /* update all voltage factors */
    for (c=panel.circuits; c!=NULL; c=c->next)
    {
        // get circuit type
        int n = (int)c->type;
        if (n<0 || n>2)
            GL_THROW("%s:%d circuit %d has an invalid circuit type (%d)", obj->oclass->name, obj->id, c->id, (int)c->type);
        c->pLoad->voltage_factor = c->pV->Mag() / ((c->pLoad->config&EUC_IS220) ? 240 : 120);
        if ((c->pLoad->voltage_factor > 1.06 || c->pLoad->voltage_factor < 0.88) && (ANSI_voltage_check==true))
            gl_warning("%s - %s:%d is outside of ANSI standards (voltage = %.0f percent of nominal 120/240)", obj->name, obj->oclass->name,obj->id,c->pLoad->voltage_factor*100);
    }
    return TS_NEVER;
}
```

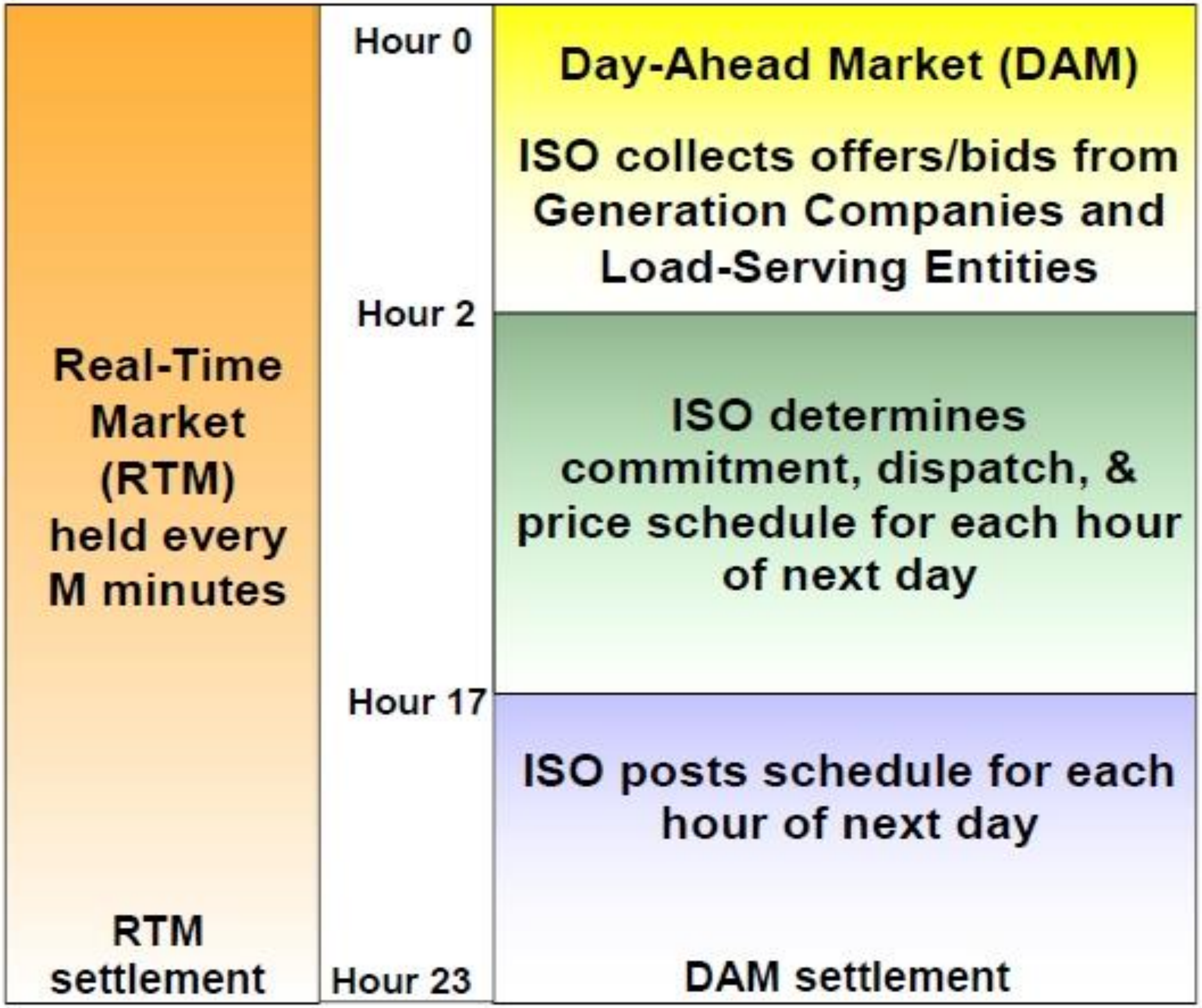
However, in the following code snippet, 240 V is hard-coded to adjust the voltage:



```
#pragma warning("house_e: add update_system voltage adjustment for heating")
    double voltage_adj = (((pCircuit_V[0]).Mag() * (pCircuit_V[0]).Mag()) / (240.0 * 240.0) * load.impedance_fraction + ((pCircuit_V[0]).Mag() / 240.0) * load.current_fraction + load.power_fraction);
    double voltage_adj_resistive = ((pCircuit_V[0]).Mag() * (pCircuit_V[0]).Mag()) / (240.0 * 240.0);
```

- ✓ Should not the nominal voltage here be set to the nominal voltage in the previous step?

Preparation for AMES V5.0 J-Unit Testing: Event Sequencing


Hour 0
Day-Ahead Market (DAM)
ISO collects offers/bids from Generation Companies and Load-Serving Entities
Hour 2
ISO determines commitment, dispatch, & price schedule for each hour of next day
Hour 17
ISO posts schedule for each hour of next day
Hour 23
DAM settlement
Real-Time Market (RTM) held every M minutes
RTM settlement

AMES V5.0: Sequence of Events During a Typical Simulated Day

AMES – PSST : Sequence of Events

Step 0: Set D = 1

Step 1: Set H = 0 and I = 0

Step 2: AMES starts DAM on day 'D' to plan for next-day operations on day 'D+1'.

Step 3: AMES writes 'ReferenceModel.dat' file and makes an external call to PSST to solve a Security-Constrained Unit Commitment (SCUC) optimization.

Step 4: PSST reads the input file 'ReferenceModel.dat' and performs SCUC.

Step 5: Status of each unit is set to 0/1 based on the SCUC outcomes in the previous step and another call to SCUC is made to obtain dual solutions – i.e LMP at each bus.

Step 6: PSST writes SCUC outcomes from Step 3 and Step 4 into 'xfertoames.dat' and 'DAMLMP.dat' files.

Step 7: AMES reads commitment and LMP data from 'DAMLMP.dat' and 'xfertoames.dat' and updates its DAM outcomes.

Step 8: AMES starts RTM operations for interval 'I' (interval duration = 'M' min).

Step 9: AMES writes 'rt-unitcommitments.dat' (contains generator unit commitments of day 'D' and 'RTReferenceModel.dat' and makes an external call to PSST to solve Security-Constrained Economic Dispatch (SCED) optimization.

AMES – PSST : Sequence of Events … Continued

Step 10: PSST writes SCED outcomes into ‘RTSCED.dat’.

Step 11: AMES reads dispatch and LMP data from ‘RTSCED.dat’ and updates RTM outcomes.

Step 12: AMES posts RTM LMPs.

Step 13:

```
Increment I;
If (I*M % 60 == 0) {
        H++;
        Reset I = 0;
  }
  If (H% 24 == 0) {
        increment D;
        goto Step 1;
  }
  Else goto Step 8.
```