Contents

Class Diagram

@image latex classDiagramr @image html classDiagram.png

IDF File

Process flow for integrating with EnergyPlus

- EnergyPlus starts, reads in the IDF, setups up the Simulation for interfacing with No-MASS
- EnergyPlus extracts the NoMASS.fmu file
- For each run period:
- Initialise No-MASS
- For each timestep:
  - Set the variables in the No-MASS by passing an array of double values and an array of the value references given in the XML file
  - Calls the No-MASS each step, No-MASS performs its calculations calculations
  - Return the results of the No-MASS back to EnergyPlus, sends an array of the value references given in the XML file and the value of variable
- Terminates No-MASS
- End of simulation

EnergyPlus Examples

Example 1: Shoe box configuration file

The following example enables No-MASS to run a non-residential building configuration.

- A building is defined
  - Two agents are working in the building and assigned to the only zone
  - Both agents have the same presence profile, have equal power and use the same shade and window model

XML File

```
<?xml version="1.0" encoding="UTF-8"?>
<simulation>
  <seed>0</seed>
  <timeStepsPerHour>12</timeStepsPerHour>
  <beginMonth>1</beginMonth>
  <endMonth>12</endMonth>
  <beginDay>1</beginDay>
  <endDay>31</endDay>
```

```
<learn>0</learn>
<buildings>
<building>
  <zone>
    <name>Block1:Zone1</name>
    <activities>IT</activities>
    <groundFloor>1</groundFloor>
    <windowCount>1</windowCount>
  </zone>
  <agents>
    <agent>
      <shade>1</shade>
      <window>1</window>
      <office>Block1:Zone1</office>
      <power>0.5</power>
      <profile>
        <monday>0.021,0.021,0.021,0.021,0.021,0.021,0.021,0.025,0.250,0.422,0.309,0.377,0
        <tuesday>0.040,0.038,0.038,0.038,0.038,0.038,0.038,0.046,0.320,0.401,0.325,0.417,0
        <wednesday>0.041,0.041,0.041,0.041,0.041,0.041,0.041,0.062,0.342,0.403,0.297,0.342
        <thursday>0.024,0.024,0.024,0.024,0.024,0.024,0.024,0.029,0.265,0.373,0.271,0.368,
        <friday>0.030,0.029,0.029,0.029,0.029,0.029,0.029,0.055,0.327,0.367,0.304,0.373,0.
        <saturday>0.028,0.030,0.029,0.029,0.030,0.029,0.029,0.030,0.030,0.030,0.035,0.037,
        <sunday>0.021,0.021,0.021,0.021,0.021,0.021,0.021,0.021,0.024,0.025,0.032,0.045,0.
      </profile>
    </agent>
    <agent>
      <shade>1</shade>
      <window>1</window>
      <office>Block1:Zone1</office>
      <power>0.5</power>
      <profile>
        <monday>0.021,0.021,0.021,0.021,0.021,0.021,0.021,0.025,0.250,0.422,0.309,0.377,0.
        <tuesday>0.040,0.038,0.038,0.038,0.038,0.038,0.038,0.046,0.320,0.401,0.325,0.417,0
        <wednesday>0.041,0.041,0.041,0.041,0.041,0.041,0.041,0.062,0.342,0.403,0.297,0.342
        <thursday>0.024,0.024,0.024,0.024,0.024,0.024,0.024,0.029,0.265,0.373,0.271,0.368,
        <friday>0.030,0.029,0.029,0.029,0.029,0.029,0.029,0.055,0.327,0.367,0.304,0.373,0.
        <saturday>0.028,0.030,0.029,0.029,0.030,0.029,0.029,0.030,0.030,0.030,0.035,0.037,
        <sunday>0.021,0.021,0.021,0.021,0.021,0.021,0.021,0.021,0.024,0.025,0.032,0.045,0.
      </profile>
    </agent>
  </agents>
</building>
</buildings>
<models>
  <presencePage>1</presencePage>
  <shades>
```

```
  <enabled>1</enabled>
  <shade>
  <id>1</id>
  <name>001-28-X</name>
  <a01arr>-6.39</a01arr>
  <b01inarr>0.00141</b01inarr>
  <b01sarr>2.17</b01sarr>
  <a10arr>-2.71</a10arr>
  <b10inarr>-0.00364</b10inarr>
  <b10sarr>-3.139</b10sarr>
  <a01int>-6.66</a01int>
  <b01inint>0.00086</b01inint>
  <b01sint>1.27</b01sint>
  <a10int>-3.87</a10int>
  <b10inint>-0.00358</b10inint>
  <b10sint>-2.683</b10sint>
  <afullraise>0.435</afullraise>
  <boutfullraise>1.95</boutfullraise>
  <bsfullraise>-0.0000231</bsfullraise>
  <bsfulllower>0.00000091</bsfulllower>
  <boutfulllower>-2.23</boutfulllower>
  <afulllower>-0.27</afulllower>
  <aSFlower>-2.294</aSFlower>
  <bSFlower>1.522</bSFlower>
  <shapelower>1.708</shapelower>
  </shade>
</shades>
<windows>
  <enabled>1</enabled>
  <window>
  <id>1</id>
  <aop>2.151</aop>
  <bopout>0.172</bopout>
  <shapeop>0.418</shapeop>
  <a01arr>-13.70</a01arr>
  <b01inarr>0.308</b01inarr>
  <b01outarr>0.0395</b01outarr>
  <b01absprevarr>1.862</b01absprevarr>
  <b01rnarr>-0.43</b01rnarr>
  <a01int>-11.78</a01int>
  <b01inint>0.263</b01inint>
  <b01outint>0.0394</b01outint>
  <b01presint>-0.0009</b01presint>
  <b01rnint>-0.336</b01rnint>
  <a01dep>-8.72</a01dep>
  <b01outdep>0.1352</b01outdep>
```

```
      <b01absdep>0.85</b01absdep>
      <b01gddep>0.82</b01gddep>
      <a10dep>-8.68</a10dep>
      <b10indep>0.222</b10indep>
      <b10outdep>-0.0936</b10outdep>
      <b10absdep>1.534</b10absdep>
      <b10gddep>-0.845</b10gddep>
      </window>
      </windows>
    <lights>1</lights>
  </models>
</simulation>
```

IDF File

```
Output:Variable,*,Schedule Value,Timestep;
Output:Variable,*,AFN Surface Venting Window or Door Opening Factor,timestep;
Output:Variable,*,Zone Exterior Windows Total Transmitted Beam Solar Radiation Rate,timestep
Output:Variable,*,Window Shading Fraction,timestep;

Output:Variable,*,Zone People Radiant Heating Rate,timestep;
Output:Variable,*,Zone Air Relative Humidity,timestep;
Output:Variable,*,Zone Mean Radiant Temperature,timestep;
Output:Variable,*,Zone People Occupant Count,timestep;
Output:Variable,*,Daylighting Reference Point 1 Illuminance,timestep;
Output:Variable,*,Zone Mean Radiant Temperature,timestep;
Output:Variable,*,Site Exterior Horizontal Sky Illuminance,timestep;
Output:Variable,*,Site Rain Status,timestep;
Output:Variable,*,Site Outdoor Air Drybulb Temperature,timestep;
Output:Variable,*,Zone Lights Electric Power,Timestep;
Output:Variable,*,Zone Lights Electric Energy,Timestep;
Output:Variable,*,Zone Air System Sensible Heating Energy,Timestep;
Output:Variable,*,Zone Air System Sensible Heating Rate,Timestep;
Output:Variable,*,Zone Air System Sensible Cooling Energy,Timestep;
Output:Variable,*,Zone Air System Sensible Cooling Rate,Timestep;
Output:Variable,*,Zone Mean Air Temperature,Timestep;
Output:Variable, *, Zone Thermal Comfort Fanger Model PMV, Timestep;
Output:Variable, *, Zone Thermal Comfort Fanger Model PPD, Timestep;


Output:Variable,*,Zone People Radiant Heating Rate,monthly;
Output:Variable,*,Zone Air Relative Humidity,monthly;
Output:Variable,*,Zone Mean Radiant Temperature,monthly;
Output:Variable,*,Zone People Occupant Count,monthly;
Output:Variable,*,Daylighting Reference Point 1 Illuminance,monthly;
Output:Variable,*,Zone Mean Radiant Temperature,monthly;
```

```
Output:Variable,*,Site Exterior Horizontal Sky Illuminance,monthly;
Output:Variable,*,Site Rain Status,monthly;
Output:Variable,*,Site Outdoor Air Drybulb Temperature,monthly;
Output:Variable,*,Zone Lights Electric Power,monthly;
Output:Variable,*,Zone Lights Electric Energy,monthly;
Output:Variable,*,Zone Air System Sensible Heating Energy,monthly;
Output:Variable,*,Zone Air System Sensible Heating Rate,monthly;
Output:Variable,*,Zone Air System Sensible Cooling Energy,monthly;
Output:Variable,*,Zone Air System Sensible Cooling Rate,monthly;
Output:Variable,*,Zone Mean Air Temperature,monthly;

Output:Variable,*,Zone People Radiant Heating Rate,runperiod;
Output:Variable,*,Zone Air Relative Humidity,runperiod;
Output:Variable,*,Zone Mean Radiant Temperature,runperiod;
Output:Variable,*,Zone People Occupant Count,runperiod;
Output:Variable,*,Daylighting Reference Point 1 Illuminance,runperiod;
Output:Variable,*,Zone Mean Radiant Temperature,runperiod;
Output:Variable,*,Site Exterior Horizontal Sky Illuminance,runperiod;
Output:Variable,*,Site Rain Status,runperiod;
Output:Variable,*,Site Outdoor Air Drybulb Temperature,runperiod;
Output:Variable,*,Zone Lights Electric Power,runperiod;
Output:Variable,*,Zone Lights Electric Energy,runperiod;
Output:Variable,*,Zone Air System Sensible Heating Energy,runperiod;
Output:Variable,*,Zone Air System Sensible Heating Rate,runperiod;
Output:Variable,*,Zone Air System Sensible Cooling Energy,runperiod;
Output:Variable,*,Zone Air System Sensible Cooling Rate,runperiod;
Output:Variable,*,Zone Mean Air Temperature,runperiod;


ExternalInterface,
  FunctionalMockupUnitImport;               !- Name of External Interface


ExternalInterface:FunctionalMockupUnitImport,
  agentFMU.fmu,                             !- FMU Filename
   15,                                      !- FMU Timeout in milli-seconds
   0;                                       !- FMU LoggingOn Value


ExternalInterface:FunctionalMockupUnitImport:From:Variable,
  Block1:Zone1,                            !- EnergyPlus Key Value,
  Zone Air Relative Humidity,              !- EnergyPlus Variable Name,
  agentFMU.fmu,                            !- FMU Filename,
  FMI,                                     !- FMU Instance Name,
  Block1:Zone1ZoneAirRelativeHumidity;     !- FMU Variable Name
```

```
ExternalInterface:FunctionalMockupUnitImport:From:Variable,
  Block1:Zone1,                             !- EnergyPlus Key Value,
  Zone Mean Radiant Temperature,            !- EnergyPlus Variable Name,
  agentFMU.fmu,                             !- FMU Filename,
  FMI,                                      !- FMU Instance Name,
  Block1:Zone1ZoneMeanRadiantTemperature;   !- FMU Variable Name

ExternalInterface:FunctionalMockupUnitImport:From:Variable,
  Block1:Zone1,                             !- EnergyPlus Key Value,
  Zone Mean Air Temperature,                !- EnergyPlus Variable Name,
  agentFMU.fmu,                             !- FMU Filename,
  FMI,                                      !- FMU Instance Name,
  Block1:Zone1ZoneMeanAirTemperature;       !- FMU Variable Name

ExternalInterface:FunctionalMockupUnitImport:From:Variable,
  Block1:Zone1,                             !- EnergyPlus Key Value,
  Daylighting Reference Point 1 Illuminance,!- EnergyPlus Variable Name,
  agentFMU.fmu,                             !- FMU Filename,
  FMI,                                      !- FMU Instance Name,
  Block1:Zone1DaylightingReferencePoint1Illuminance;!- FMU Variable Name

ExternalInterface:FunctionalMockupUnitImport:To:Schedule,
  Block1:Zone1NumberOfOccupants,            !- EnergyPlus Key Value,
  Any Number,                               !- Schedule Type Limits Names,
  agentFMU.fmu,                             !- FMU Filename,
  FMI,                                      !- FMU Model Name,
  Block1:Zone1NumberOfOccupants,            !- FMU Model Variable Name,
  0.0;                                      !- Initial Value

  ExternalInterface:FunctionalMockupUnitImport:To:Schedule,
    Block1:Zone1Heating,                      !- EnergyPlus Key Value,
    Any Number,                               !- Schedule Type Limits Names,
    agentFMU.fmu,                             !- FMU Filename,
    FMI,                                      !- FMU Model Name,
    Block1:Zone1Heating,                      !- FMU Model Variable Name,
    0.0;                                      !- Initial Value


ExternalInterface:FunctionalMockupUnitImport:To:Actuator,
  Block1:Zone1_Wall_5_0_0_0_0_0_Win_Shading_Fraction, !- EnergyPlus Variable Name
  Block1:Zone1_Wall_5_0_0_0_0_0_Win, !- Actuated Component Unique Name
  Window Shading Fraction, !- Actuated Component Type
  Control Fraction, !- Actuated Component Control Type
  agentFMU.fmu, !- FMU Filename
  FMI, !- FMU Model Name
  Block1:Zone1BlindFraction, !- FMU Model Variable Name
```

```
  1.0; !- Initial Value!

ExternalInterface:FunctionalMockupUnitImport:To:Schedule,
  Block1:Zone1LightState,                  !- EnergyPlus Key Value,
  Any Number,                              !- Schedule Type Limits Names,
  agentFMU.fmu,                            !- FMU Filename,
  FMI,                                     !- FMU Model Name,
  Block1:Zone1LightState,                  !- FMU Model Variable Name,
  0.0;                                     !- Initial Value

ExternalInterface:FunctionalMockupUnitImport:To:Schedule,
  Block1:Zone1WindowState0,                 !- EnergyPlus Key Value,
  Any Number,                              !- Schedule Type Limits Names,
  agentFMU.fmu,                            !- FMU Filename,
  FMI,                                     !- FMU Model Name,
  Block1:Zone1WindowState0,                 !- FMU Model Variable Name,
  0.0;                                     !- Initial Value


ExternalInterface:FunctionalMockupUnitImport:To:Schedule,
  Block1:Zone1AverageGains,                !- EnergyPlus Key Value,
  Any Number,                              !- Schedule Type Limits Names,
  agentFMU.fmu,                            !- FMU Filename,
  FMI,                                     !- FMU Model Name,
  Block1:Zone1AverageGains,                !- FMU Model Variable Name,
  0.0;                                     !- Initial Value

ExternalInterface:FunctionalMockupUnitImport:From:Variable,
  Environment,                             !- EnergyPlus Key Value,
  Site Exterior Horizontal Sky Illuminance,!- EnergyPlus Variable Name,
  agentFMU.fmu,                            !- FMU Filename,
  FMI,                                     !- FMU Instance Name,
  EnvironmentSiteExteriorHorizontalSkyIlluminance;!- FMU Variable Name

ExternalInterface:FunctionalMockupUnitImport:From:Variable,
  Environment,                             !- EnergyPlus Key Value,
  Site Rain Status,                        !- EnergyPlus Variable Name,
  agentFMU.fmu,                            !- FMU Filename,
  FMI,                                     !- FMU Instance Name,
  EnvironmentSiteRainStatus;               !- FMU Variable Name

ExternalInterface:FunctionalMockupUnitImport:From:Variable,
  Environment,                             !- EnergyPlus Key Value,
  Site Outdoor Air Drybulb Temperature,    !- EnergyPlus Variable Name,
  agentFMU.fmu,                            !- FMU Filename,
  FMI,                                     !- FMU Instance Name,
```

```
  EnvironmentSiteOutdoorAirDrybulbTemperature;!- FMU Variable Name
```

Model Description File

```
<fmiModelDescription description="Model with interfaces for media with moist air that will b
  <TypeDefinitions>
    <Type name="Modelica.Blocks.Interfaces.RealInput">
      <RealType />
    </Type>
    <Type name="Modelica.Blocks.Interfaces.RealOutput">
      <RealType />
    </Type>
  </TypeDefinitions>
  <DefaultExperiment startTime="0.0" stopTime="1.0" tolerance="1E-005" />
  <ModelVariables>
    <ScalarVariable causality="input" name="Block1:Zone1ZoneAirRelativeHumidity" valueRefere
      <Real declaredType="Modelica.Blocks.Interfaces.RealInput" start="0.0" />
    </ScalarVariable>
    <ScalarVariable causality="input" name="Block1:Zone1ZoneMeanRadiantTemperature" valueRef
      <Real declaredType="Modelica.Blocks.Interfaces.RealInput" start="0.0" />
    </ScalarVariable>
    <ScalarVariable causality="input" name="Block1:Zone1ZoneMeanAirTemperature" valueReferen
      <Real declaredType="Modelica.Blocks.Interfaces.RealInput" start="0.0" />
    </ScalarVariable>
    <ScalarVariable causality="input" name="Block1:Zone1DaylightingReferencePoint1Illuminanc
      <Real declaredType="Modelica.Blocks.Interfaces.RealInput" start="0.0" />
    </ScalarVariable>
    <ScalarVariable causality="output" name="Block1:Zone1NumberOfOccupants" valueReference="
      <Real declaredType="Modelica.Blocks.Interfaces.RealOutput" start="0.0" />
    </ScalarVariable>
    <ScalarVariable causality="output" name="Block1:Zone1AverageGains" valueReference="6">
      <Real declaredType="Modelica.Blocks.Interfaces.RealOutput" start="0.0" />
    </ScalarVariable>
    <ScalarVariable causality="input" name="EnvironmentSiteExteriorHorizontalSkyIlluminance"
      <Real declaredType="Modelica.Blocks.Interfaces.RealInput" start="0.0" />
    </ScalarVariable>
    <ScalarVariable causality="input" name="EnvironmentSiteRainStatus" valueReference="8">
      <Real declaredType="Modelica.Blocks.Interfaces.RealInput" start="0.0" />
    </ScalarVariable>
    <ScalarVariable causality="input" name="EnvironmentSiteOutdoorAirDrybulbTemperature" val
      <Real declaredType="Modelica.Blocks.Interfaces.RealInput" start="0.0" />
    </ScalarVariable>
    <ScalarVariable causality="input" name="EMSwarmUpComplete" valueReference="10">
      <Real declaredType="Modelica.Blocks.Interfaces.RealInput" start="0.0" />
    </ScalarVariable>
```

```
    <ScalarVariable causality="input" name="EMSepTimeStep" valueReference="11">
      <Real declaredType="Modelica.Blocks.Interfaces.RealInput" start="0.0" />
    </ScalarVariable>
    <ScalarVariable causality="output" name="Block1:Zone1BlindFraction" valueReference="12">
      <Real declaredType="Modelica.Blocks.Interfaces.RealInput" start="1.0" />
    </ScalarVariable>
    <ScalarVariable causality="output" name="Block1:Zone1WindowState0" valueReference="13">
      <Real declaredType="Modelica.Blocks.Interfaces.RealInput" start="0.0" />
    </ScalarVariable>
    <ScalarVariable causality="output" name="Block1:Zone1LightState" valueReference="14">
      <Real declaredType="Modelica.Blocks.Interfaces.RealInput" start="0.0" />
    </ScalarVariable>
  </ModelVariables>
</fmiModelDescription>
```

Functional Mockup Unit (FMU)

A Functional Mockup Unit is a program that allows EnergyPlus to couple new custom functionality to EnergyPlus. It is a generic method for allowing simulation programs to be extended. When applied information can be passed from EnergyPlus at runtime to a FMU which can then make calculations and send the results back to EnergyPlus. These results can then be used in EnergyPlus future calculations. For more information on FMUs please see the EnergyPlus website

##What is a FMU

An FMU consists of three main parts: * FMU Library (Windows DLL) * Model Description File * IDF Code used to enable the FMU

##How does an FMU Work

An FMU works using a simple data exchange methodology. At each timestep EnergyPlus sends a set of predefined variables (x) to the FMU (e.g. Zone Mean Air Temperature and/or Site Rain status...). The FMU performs a calculation on the values and returns a set of results (y) to EnergyPlus (e.g. occupant location and/or shade status...). These return values can be made to overwrite EnergyPlus values for the next timestep.

@image latex image7 @image html image7.png

A FMU can be developed to simulate heating setpoints, HVAC controllers or extend EnergyPlus to include any missing functionality.

##Linking to EnergyPlus

To run a simulation you will also need to run EnergyPlus with the idf file, epw file and the FMU in the same directory

##Model description

The model description tab allows you to specify the variables that will be passed between the FMU and EnergyPlus. EnergyPlus and the FMU use this to know what values they should expect and what they should send in return. This is written in XML format, see below.

@image latex image1 @image html image1.png

The name of the the FMU(1) and the name of the model(2) need to be kept consistent throughout the dialog and should be provided with the imported FMU. The GUID(2) is a unique string that will also be provided with the FMU if required. The variables to be pass are specified between items 4 and 11. A variable is defined with items 5 to 10. The start of the variable contains the the name associated to it, a unique number and direction the value will be sent. All EnergyPlus variables are of "Real" value so the type of variable will always be real. The name of the variable will correspond to the value given in IDF value under FMU Model Variable Name definition, these should be identical. The value reference should always be unique to the variable and is used by EnergyPlus and the FMU to keep track of the values passed. The causality of the variable specifies the direction the value of the variable will be sent. Shown below:

@image latex image8 @image html image8.png

Using this XML file you will also have to define in the IDF file which values EnergyPlus needs to send.

##IDF Script

The IDF script contents for FMU is explained in the EnergyPlus documentation, see here

The IDF FMU details tell EnergyPlus the values that it needs to know in order to run the FMU program. First we define a ExternalInterface:FunctionalMockupUnitImport which specifies the file name of the FMU in the EnergyPlus folder. This is the name of the FMU plus the ".fmu" file extension.

@image latex image6 @image html image6.png

Next we define all the variables that EnergyPlus will send to the FMU as (2). This includes the variable name and key (3), the name of the FMU(4), the model name (5), and the FMU variable name (6). The values for 5 & 6 need to be kept consistent with the values given in the corresponding variables in the model description file.

Finally there is the returned value from the FMU, in this case the shading value as an actuator. These are the available external types that can be written to: * ExternalInterface:FunctionalMockupUnitImport:To:Variable

* ExternalInterface:FunctionalMockupUnitImport:To:Schedule * ExternalInterface:FunctionalMockupUnitImport:To:Actuator

##Developing a FMU

Developing an FMU requires some advance computer science knowledge, first how to program and second how to build that program into a library exported as a DLL in Microsoft Windows. More information about developing FMUs can be found on the Functional Mockup Interface website.

The basic flow for an FMU is as followed:

@image latex image3 @image html image3.png

- EnergyPlus starts, reads in the IDF, setups up the Simulation for FMU usage
- EnergyPlus extracts the .fmu file
- For each run period:
  - Initialise the FMU model by calling the initialise method
  - For each timestep:
    - Set the variables in the FMU by passing an array of double values and an array of the value references given in the XML file
    - Calls the FMU do step method, the fmu does any calculations
    - Return the results of the FMU back to EnergyPlus, sends a preallocated array that is written to and an array of the value references given in the XML file
  - Terminate the FMU by calling the terminate method
- End of simulation

##FMU file format

A fmu file such as agentFMU.fmu is just a zipped folder renamed with the from agentFMU.zip to agentFMU.fmu. The zipped folder has the following folder format:

@image latex image4 @image html image4.png

EnergyPlus extracts these files so the model description file can be edited to work with different buildings and configurations. To Debug the DLL with EnergyPlus for testing this format will need to be used. Copy the DLL to the correct binaries folder, zip the folder and copy the renamed fmu to the EnergyPlus folder. Have Visual Studio run EnergyPlus, If everything is correctly setup then Visual Studio will be able to stop the program at breakpoints for inspection.

Implementation

When launched, No-MASS first builds an agent population, assigning a profile to each member dependent on the input parameters supplied in a No-MASS configuration file. These profiles influence the models predicting activities (e.g. sleeping,

bathing, watching TV) and dependent behaviours (e.g. opening windows or lowering shading devices). Once the agent profiles are assigned No-MASS proceeds according to one of two scenarios. If the building is non-residential, chains of presence and absence are calculated; otherwise for residential buildings chains of activities are calculated, and the corresponding activity locations are assigned to each activity. Communication with the building solver is then performed. The building solver that we have coupled with No-MASS is EnergyPlus. To achieve this we use the generic Functional Mockup Interface (FMI) co-simulation standard, so that we can also couple No-MASS with any other FMI compliant BPS software. The building solver calculates the environmental conditions within the zones that the agents have been allocated to and parses the results to No-MASS via the FMI. Each agent then retrieves its pre-processed location or presence (Page 2008). State parameters are then set based on the activity (Jaboob 2015) that is performed, affecting the agents' clothing level, metabolic rate etc. We then calculate the metabolic gains of each individual occupant. Next we call models predicting the agents' use of shades (Haldi 2010), windows (Haldi 2009), lights (Reinhart 2004) and heating system setpoints. To facilitate the modelling of agent social interactions we wrap these models in a social interaction framework; emulating negotiations through a vote casting mechanism. The prediction of agents' heating setpoint choices is based on a reinforcement learning model, allowing agents to learn over a number of simulation replicates the setpoints that best maintain their PMV within acceptable bounds. BDI rules are included to model relatively straight forward interactions for which data is scarce, such as closing shades for privacy in the home while showering. Finally the results are parsed back to the building solver which calculates the environmental conditions arising from the modelled interactions at the next time step.

No-MASS was built from the ground up, C++ was chosen as the development language as it is simple to integrate with EnergyPlus (Crawley 2001), our chosen building simulation tool is also developed in C++. Using the same language allows for easy communication between the two tools. EnergyPlus developed by the US Department of Energy, is well tested, well documented and open source; allowing us to readily understand how to connect to it. There are also two interfaces that allow other tools to interact with it, without altering the EnergyPlus source code. The first is through the building controls virtual test bed (BCVTB) and the second is through the Functional Mockup Interface (FMI) (Nouidui 2014). The No-MASS platform connects to EnergyPlus using FMI, which is an open standard so that No-MASS could in principle be integrated with any other FMI compliant simulation tool. This is chosen over the BCVTB as it allows direct communication through C++ double precision arrays using predefined calling points, whereas BCVTB requires calls over sockets adding complexity and slowing the processing time. The calling points are well documented, with No-MASS only using the initialise function, the receive an array of doubles function for the environmental variables and the send an array of doubles function for the occupant interactions. The array of values that No-MASS receives at each

time step is defined in the XML file ModelDescription.xml. At the beginning of the time step the following environmental variables are received: horizontal sky illuminance, rain status, outdoor air dry-bulb temperature, zone air temperature, zone humidity, indoor radiant temperature and indoor illuminance. Returned to EnergyPlus are the number of occupants in a zone, their metabolic gains, appliance gains, the window status, the blind shading fraction, the lighting status and the heating setpoint. Due to the window, shading and location/presence models used within No-MASS a sub-hourly timestep is recommended (ie. 5 minutes), as longer timesteps may overestimate the implication of the occupant interactions. For example the response time to an agent opening a window may be short with the room cooling in just a few minutes. An agent can only respond at the next time step, if the timesteps are not sufficiently short in length, the open window may over cool the room.

@image latex FlowDiagramExistingModelsIncluded @image html FlowDiagramExistingModelsIncluded.png

Model Description

[TOC]

The model description tab allows you to specify the variables that will be passed between the FMU and EnergyPlus. EnergyPlus and the FMU use this to know what values they should expect and what they should send in return.

This is written in XML format, see below.

```
<fmiModelDescription description="Model with interfaces for media with moist air that will b
  <TypeDefinitions>
    <Type name="Modelica.Blocks.Interfaces.RealInput">
      <RealType />
    </Type>
    <Type name="Modelica.Blocks.Interfaces.RealOutput">
      <RealType />
    </Type>
  </TypeDefinitions>
  <DefaultExperiment startTime="0.0" stopTime="1.0" tolerance="1E-005" />
  <ModelVariables>
    <ScalarVariable causality="input" name="Block1:Zone1ZoneAirRelativeHumidity" valueRefere
      <Real declaredType="Modelica.Blocks.Interfaces.RealInput" start="0.0" />
    </ScalarVariable>
    <ScalarVariable causality="input" name="Block1:Zone1ZoneMeanRadiantTemperature" valueRef
      <Real declaredType="Modelica.Blocks.Interfaces.RealInput" start="0.0" />
    </ScalarVariable>
    <ScalarVariable causality="output" name="Block1:Zone1BlindFraction" valueReference="3">
      <Real declaredType="Modelica.Blocks.Interfaces.RealInput" start="1.0" />
    </ScalarVariable>
```

```
  ...
  </ModelVariables>
</fmiModelDescription>
```

ScalarVariable

ScalarVariable

A ScalarVariable declares the parameter that will be passed

Name

Name

The name of the variable gives the program understanding as to the parameter being passed, NoMass expects these to be declared in a specific way depending on how they are used by a model.

ValueReference

ValueReference

The value reference should be unique and links the variable in the code to the specific value. For example, the name will be linked to a the value reference so when the variable is passed from the main program it is passed with its value reference. With this information the variable is known.

Causality

Causality

The causality of the variable specifies the direction the value of the variable will be sent.

- Input is into the FMU
- Output is out of the FMU into the main program

Real

Real

For NoMASS the variable is always a real.

declaredType

declaredType The declared type is always Modelica.Blocks.Interfaces.RealInput.

start

start Start is the initial value the variable will have.

Simulation Configuration

[TOC]

This is an XML document that is used to inform No-MASS how to run.

It contains the information for the simulation period, buildings, zones, agents and appliances.

The initial tag to define a number simulation ~~~~ ... ~~~~

Seed

Seed

As we use stochastic models we define a seed value. This can be removed and No-MASS will generate its own. However as it is often required for results can be repeated, doing it manully will allow the random values to be the same.

```
<seed>0</seed>
```

Time Period

Time Period

Here we define the time period information. ~~~~ 60 1 1 1 7 ~~~~

Saving Results

Saving Results

To save the results enable the save tag with the value 1. This writes out the No-MASS results to a NoMASS.out file. However this can slow the simulation down and can therefore be disabled with a 0 value. ~~~~ 1 ~~~~

The results file can be further filtered using regular expressions like that of below. Any output names that matches one of the options below is saved. If an output name does not match below it is discarded.

```
<output>
  <regex>nsim</regex>
  <regex>TimeStep</regex>
  <regex>hour</regex>
  <regex>day</regex>
  <regex>Building[0-9]\d?_Appliance[0-9]\d?_received.*</regex>
  <regex>Building[0-9]\d?_Appliance1[0-9]\d?_received.*</regex>
  <regex>Building[0-9]\d?_Appliance10_supplied.*</regex>
```

```
  <regex>Building[0-9]\d?_Appliance10_supplied</regex>
  <regex>Building[0-9]\d?_Appliance1[0-9]\d?_supplied.*</regex>
  <regex>Building[0-9]\d?_Appliance1[0-9]\d?_received.*</regex>
  <regex>Building[0-9]\d?_Appliance[0-9]\d?_previous_state</regex>
  <regex>Building[0-9]\d?_Appliance[0-9]\d?_action</regex>
  <regex>Building[0-9]\d?_Appliance[0-9]\d?_reward</regex>
  <regex>Building[0-9]\d?_Appliance[0-9]\d?_state</regex>
  <regex>Building[0-9]\d?_Appliance1_cost</regex>
  <regex>Building[0-9]\d?_Appliance4_cost</regex>
  <regex>grid_power</regex>
  <regex>grid_cost</regex>
</output>
```

Q-Learning

Q-Learning for Occupants

Q-Learning occupant model specific however there will be times when you want to read from the learning data but not update the data with new values, for example once the training period is over. Use learnupdate to turn this off. The learn ep value is the epsilon value of Q-Learning and can be altered here with a double value. ~~~~ 1 0.01 ~~~~

Grid cost

The cost of using the grid

Any appliance that using energy from the grid will use a tariff the is multiplied by the power used. This is specified per day/hour/halfhour using an array with a length of 1/24/44 respectively. Each value is the cost of one unit of power.

0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.1, 0.9, 0.9, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.5, 0.9, 0.9, 0.9, 0.9, 0.9, 0.9, 0.5, 0.5, 0.1

Buildings

Buildings

Buildings are defined within the buildings tag, like so. ~~~~ ... ... ~~~~

Building

Building

A Building is define with a set of zones, agents and appliances.

```
<building>
      <zone>...</zone>
```

```
      <zone>...</zone>
      <Appliances>...</Appliances>
      <agents>...</agents>
</building>
```

Zone (Rooms)

Zone (Rooms)

A zone is a section of building often there is one per room, although a room can be sub divide into individual zones. Each zone is given a name, a set of activities and the number of windows in the the zone.

There are 10 activities as defined by the activity model: * Cooking * Cleaning * AudioVisual * Metabolic * Passive * Washing * WashingAppliance * Sleep * IT * Out

These are given to a zone and are comma seperated like below, however Out does not need to be defined.

The activities assign an agent to a zone, for example if Cooking is defined as an activity for that zone when an agent is in the state cooking they are also defined to this zone.

```
<zone>
    <name>Block1:Kitchen</name>
    <activities>Cooking,Cleaning</activities>
    <windowCount>1</windowCount>
</zone>
<zone>
    <name>Block1:LivingRoom</name>
    <activities>AudioVisual,Metabolic,Passive</activities>
    <windowCount>1</windowCount>
</zone>
```

Appliances

Appliances

There are the following types of appliance:

- Large
- LargeCSV
- LargeLearning
- LargeLearningCSV
- Small
- PV
- CSV
- FMI

- Grid
- Battery
- BatteryGridCost

```
<Appliances>
  <Large>...</Large>
  <LargeCSV>...</LargeCSV>
  <LargeLearning>...</LargeLearning>
  <LargeLearningCSV>...</LargeLearningCSV>
  <Small>...</Small>
  <pv>...</pv>
  <FMI>...</FMI>
  <Grid>...</Grid>
  <Battery>...</Battery>
  <BatteryGridCost>...</BatteryGridCost>
</Appliances>
```

Appliance Priority

Appliance Priority

Each Appliance is given a priority, which is a value that defines the order in which the device will receive its demanded energy, higher priority energy receive there demand first.

Large Appliances

Large Appliances

The large appliance has a id that relates to the id of the large appliances defined in the ApplianceLarge.xml file.

```
<Large>
  <id>0</id>
  <priority>0</priority>
</Large>
```

The large appliance can also be given a set of activities meaning that the appliance will only turn on when an agent is in the activity specified.

```
<Large>
  <id>0</id>
  <priority>0</priority>
  <activities>Cooking,Cleaning</activities>
</Large>
```

If required it is possible to override the large appliance predicted power profile with that of a profile specified through a csv file.

```
<LargeCSV>
```

```
  <id>1</id>
  <priority>0.1,0.1,0.1,0.1,0.1,0.1,0.1, 0.1  ,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0
  <activities>WashingAppliance</activities>
  <fileProfile>profile1.csv</fileProfile>
</LargeCSV>
```

Large Appliances Q-Learning

Large Appliances Q-Learning

Define in the same way as a large appliance only now we use the tag LargeLearning instead. This allows the applaince to shift its profile over time. Using the Q-Learning algorithm hopefully to a time when the supplied power is cheap.

```
<LargeLearning>
  <id>1</id>
  <priority>0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,
  <timeRequired>0.1,0.1,0.1,0.1,0.1,0.1,0.1,1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.
  <epsilon>0.1</epsilon>
  <alpha>0.3</alpha>
  <gamma>0.1</gamma>
  <updateQTable>1</updateQTable>
  <activities>WashingAppliance</activities>
</LargeLearning>
```

If required it is possible to override the large learning appliance predicted power profile with that of a profile specified through a csv file.

```
<LargeLearningCSV>
  <id>4</id>
  <priority>0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,
  <timeRequired>0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,0.1,
  <epsilon>0.1</epsilon>
  <alpha>0.3</alpha>
  <gamma>0.1</gamma>
  <updateQTable>1</updateQTable>
  <activities>WashingAppliance</activities>
  <fileProfile>profile4.csv</fileProfile>
</LargeLearningCSV>
```

Small Appliances

Small Appliances

The small appliance are defined through csv files included with NoMASS

```
<Small>
  <id>6</id>
```

```
  <priority>6</priority>
  <WeibullParameters>weibull_parameters_audiovisual.csv</WeibullParameters>
  <StateProbabilities>state_probability_audiovisual.csv</StateProbabilities>
  <Fractions>mean_fractional_audiovisual.csv</Fractions>
  <SumRatedPowers>sum_rated_av.txt</SumRatedPowers>
</Small>
```

PV

PV

Filename is the CSV profile used to generate the supply. Cost is the how much the supplied power is.

```
<pv>
  <id>10</id>
  <priority>0</priority>
  <filename>PVBowler2013_365.csv</filename>
  <cost>0.02</cost>
</pv>
```

CSV Appliance

CSV Appliance

Filename is the CSV profile used to generate the supply/demand. Cost is the how much the supplied power is. ~~~~ 11 100 HeatingPower.csv turbineSupply.csv 0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.0,0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.2,0.2 ~~~~

FMI Appliance

FMI Appliance

Only used for appliances from the FMI interface (such as EnergyPlus sending the power), the variable name needs to correspond with the name in the modeldescription.xml file.

```
<FMI>
  <id>11</id>
  <priority>0</priority>
  <variablename>HVACPower</variablename>
</FMI>
```

Grid

Grid

Calculate the supply needed based on supply of the other appliances minus the demand of the other appliances. Cost is the how much the supplied power is.

```
<Grid>
  <id>1000</id>
  <priority>0</priority>
  <cost>1.00</cost>
</Grid>
```

Battery

Battery

The Battery can learn when discharge it stored energy. This is achieved through the q-learning algorithm. With the battery it is possible to define when it can charge and discharge to the local neighbourhood.

```
<battery>
  <id>12</id>
  <priority>0</priority>
  <epsilon>0.1</epsilon>
  <alpha>0.3</alpha>
  <gamma>0.1</gamma>
  <updateQTable>1</updateQTable>
  <batteryneighbourhooddischarge>0</batteryneighbourhooddischarge>
  <batteryneighbourhoodcharge>1</batteryneighbourhoodcharge>
</battery>
```

Agents

Agents

The definition of an agent. The window and shade parameter is the id of the window and shade profile that the agent uses. Office and bedroom are the corresponding locations for acticities IT and Sleep. Profile is the activity profile to use, in the example we specify the file used. The other parameters correspond to the activity model.

```
<agents>
    ...
      <agent>
        <shade>1</shade>
        <window>1</window>
        <office>Block1:Zone1</office>
```

```
        <power>0.5</power>
        <age>age2</age>
        <computer>computer0</computer>
        <civstat>civstat1</civstat>
        <unemp>unemp0</unemp>
        <retired>retired1</retired>
        <edtry>edtry1</edtry>
        <famstat>famstat3</famstat>
        <sex>sex2</sex>
        <bedroom>Block2:MasterBedroom</bedroom>
        <profile>
          <file>Activity.xml</file>
        </profile>
      </agent>
      ...
</agents>
```

models

models

External to the the building tag are the model tags these specify the possible window or shading models to use. As defined on a per agent basis using ids. ~~~~ ... 0 1 ... ... 0 1 ... ... ~~~~

shades

shades

These are the coefficients as defined by the shading model

```
<shade>
  <id>23</id>
  <name>204-10</name>
  <a01arr>-6.17</a01arr>
  <b01inarr>0.00114</b01inarr>
  <b01sarr>2.17</b01sarr>
  <a10arr>1.2</a10arr>
  <b10inarr>-0.00674</b10inarr>
  <b10sarr>-3.139</b10sarr>
  <a01int>-7.67</a01int>
  <b01inint>0.00088</b01inint>
  <b01sint>1.27</b01sint>
  <a10int>-2.61</a10int>
  <b10inint>-0.0051</b10inint>
  <b10sint>-2.683</b10sint>
  <afullraise>0.435</afullraise>
```

```
  <boutfullraise>1.95</boutfullraise>
  <bsfullraise>-0.0000231</bsfullraise>
  <bsfulllower>0.00000091</bsfulllower>
  <boutfulllower>-2.23</boutfulllower>
  <afulllower>-0.27</afulllower>
  <aSFlower>-2.294</aSFlower>
  <bSFlower>1.522</bSFlower>
  <shapelower>1.708</shapelower>
</shade>
```

shades

shades

These are the coefficients as defined by the window model.

```
<window>
  <id>1</id>
  <aop>2.151</aop>
  <bopout>0.172</bopout>
  <shapeop>0.418</shapeop>
  <a01arr>-13.88</a01arr>
  <b01inarr>0.312</b01inarr>
  <b01outarr>0.0433</b01outarr>
  <b01absprevarr>1.862</b01absprevarr>
  <b01rnarr>-0.45</b01rnarr>
  <a01int>-12.23</a01int>
  <b01inint>0.281</b01inint>
  <b01outint>0.0271</b01outint>
  <b01presint>-0.000878</b01presint>
  <b01rnint>-0.336</b01rnint>
  <a01dep>-8.75</a01dep>
  <b01outdep>0.1371</b01outdep>
  <b01absdep>0.84</b01absdep>
  <b01gddep>0.83</b01gddep>
  <a10dep>-8.54</a10dep>
  <b10indep>0.213</b10indep>
  <b10outdep>-0.0911</b10outdep>
  <b10absdep>1.614</b10absdep>
  <b10gddep>-0.923</b10gddep>
 </window>
```

Dependencies Of NoMASS

No-MASS depends on two external libraries, Google Tests and RapidXML. Google test is only needed to run the No-MASS tests.

These should be downloaded automatically with the command:

```
git submodule update --init --recursive
```

However if this fails then they can be manually downloaded.

- Google Tests
- RapidXML

Using No-MASS

There are number of simulation files that need to be considered depending on what you wish to achieve with NoMASS:

- The main file is the @ref SimulationConfig
- For interfacing with FMI the @ref ModelDescription
- For interfacing with EnergyPlus the @ref EnergyPlus

No-MASS

[TOC]

Introduction

Introduction

The No-MASS framework integrates existing models of occupant interaction and appliance usage into a tool that can be coupled with building or urban energy performance simulation tools, or run independently.

Pages

Pages

- @subpage Implementation
- @subpage dependencies
- @subpage compile
- @subpage input
- @subpage scripts
- @subpage ClassDiagram
- @subpage FMU
- @subpage EnergyPlusExamples

What is No-MASS?

What is No-MASS?

A multi-agent stochastic simulation of occupants and appliances for use with building or urban scale simulations tools. No-MASS using the generic FMI interface so that it can interface with any tool that meets the standard.

What kind of data can I get from No-MASS?

What kind of data can I get from No-MASS?

- Stochastic window openings predictions
- Stochastic external shade fractions
- Stochastic lighting predictions
- Stochastic occupant activity predictions
- Stochastic occupant presence predictions
- Stochastic large appliance use
- Stochastic small appliance use
- Machine learning appliance profile shifting
- Machine learning heating/ cooling setpoint predictions
- Beleif-Desire-Intent rules of occupant interactions

Background

Background

The resources needed to sustain the world's ever expanding population are reaching new levels. It is therefore important to reduce global energy use arising from the contribution of fossil fuels and the associated emission of greenhouse gasses. Building performance simulation is used to model the flows of energy in buildings and their systems at the design/ retrofit stage, to ensure they not only meet the demands of the occupants but also allow designers to test strategies for improving building performance.

Although a powerful building (re-)design decision support tool, building performance simulations can be subject to limitations. Studies have found that buildings may use twice as much energy as predicted at design stage. Predicted building performance continues to deviate – sometimes considerably – from that which is observed post-build. With the objective of addressing these limitations, and following the suggestions of Robinson (2011), our approach is to use multi-agent stochastic simulation for modelling occupant behaviour; to combine stochastic models into a single package that can be used to support building and urban performance simulation using a range of software.

A multi-agent simulation framework has been developed, which we have named Nottingham Multi-Agent Stochastic Simulation (No-MASS). The No-MASS framework integrates existing stochastic models of occupant interaction into a

tool that can be coupled with building or urban energy performance simulation tools. The coupling allows for simulated occupants to make changes within the simulated building environment and receive responses arising from the effects of the interactions which may stimulate future interactions. Current stochastic models do not cover all of the energy related behaviours of occupants, therefore a belief-desire-intent (BDI) rule system is used to model other interactions, supplementing the data-driven stochastic models. For example switching off the light during sleep. Agents can have unique desires causing changes within the environment that may be in conflict with the desires of other agents. To solve this problem an agent social interaction model is developed to govern the interactions between agents. For more complex interactions where BDI rules would be difficult to design, agent machine learning techniques are used, allowing the agent to learn how to respond to different stimuli.

Scripts For NoMASS

There is a NoMASS.py script that implements an interface for running No-MASS as a standalone program. The NoMASS class can be called like so.

```
from NoMASS import NoMASS
nomass = NoMASS()

nomass.runLocation = "../FMU/build/Simulation/"
nomass.locationOfNoMASS = "../FMU/build/"
nomass.configurationDirectory = home +"/Dropbox/DSM_ABM/Simulations/Configurations/WinterWee
nomass.resultsLocation = "../FMU/build/Results/"
nomass.printInput = True
nomass.numberOfSimulations = 1
nomass.learnUpToSimulation = 0
nomass.simulate()
nomass.deleteLearningData()
```

runLocation

NoMASS runs stochastically and often runs over multiple replicates.