## Economizer Opening Stuck

### Description

def description

return "Stuck dampers associated with economizers can be caused by seized actuators, broken linkages, economizer control system failures, or the failure of sensors that are used to determine damper position. In extreme cases, dampers stuck at either 100% open or closed can have a serious impact on system energy consumption or occupant comfort in the space. This fault is categorized as a fault that occur in the economizer system (damper) during the operation stage. This fault measure is based on a physical model where certain parameter(s) is changed in EnergyPlus to mimic the faulted operation; thus simulates a stuck damper by modifying the Controller:OutdoorAir object in EnergyPlus. The fault intensity (F) for this fault is defined as the ratio of economizer damper at the stuck position (0 = fully closed, 1 = fully open)."

end

### Modeler Description

def modeler\_description

return "To use this fault measure, user should choose the economizer getting faulted, the schedule of fault prevalence when to impose fault during the simulation and the damper stuck position. If a schedule of fault prevalence is not given, the model will apply the fault to the entire simulation."

end

### Measure Type

OpenStudio Measure

**Taxonomy**

HVAC.HVAC Controls

### Arguments

def arguments(model)

args = OpenStudio::Ruleset::OSArgumentVector.new

#make choice arguments for economizers

controlleroutdoorairs = model.getControllerOutdoorAirs

chs = OpenStudio::StringVector.new

controlleroutdoorairs.each do |controlleroutdoorair|

chs << controlleroutdoorair.name.to\_s

end

chs << $allchoices

econ\_choice = OpenStudio::Ruleset::OSArgument::makeChoiceArgument('econ\_choice', chs, true)

econ\_choice.setDisplayName("Choice of economizers. If you want to impose the fault on all economizers, choose #{$allchoices}")

econ\_choice.setDefaultValue($allchoices)

args << econ\_choice

#give a choice to choose schedules. If checked, the model will look up the chosen schedule for

#a schedule of fault presence and set the damper position during non-zero period as damper\_pos.

#Otherwise, the damper position entered at damper\_pos will be applied to the economizer for

#the entire simulation period

schedule\_exist = OpenStudio::Ruleset::OSArgument::makeBoolArgument('schedule\_exist', false)

schedule\_exist.setDisplayName('Check if a schedule of fault presence is needed, or uncheck to apply the fault for the entire simulation.')

schedule\_exist.setDefaultValue(false)

args << schedule\_exist

#choice of schedules for the presence of fault. 0 for no fault and other numbers means fault

args << fractional\_schedule\_choice(model)

#make a double argument for the damper position

#it should range between 0 and 1. 0 means completely closed damper

#and 1 means fully opened damper

damper\_pos = OpenStudio::Ruleset::OSArgument::makeDoubleArgument('damper\_pos', false)

damper\_pos.setDisplayName('The position of damper indicated between 0 and 1. If it is -1 and a schedule of fault prevalence is not given, the fault model will not be imposed to the building simulation without warning.')

damper\_pos.setDefaultValue(0.5) #default position 50% open

args << damper\_pos

return args

end

### Initial Condition

#Select economizer object that is being faulted.

runner.registerInitialCondition("Fixing #{econ\_choice} damper position to #{damper\_pos}")

### Final Condition

#Impose stuck damper position on the economizer object.

runner.registerFinalCondition("Damper position at #{econ\_choice} is fixed at #{damper\_pos}")

### Not Applicable

#When the Economizer Control Type is defined as NoEconomizer,

runner.registerAsNotApplicable("#{econ\_choice} does not have an economizer. Skipping......")

### Warning

n/a

### Error

#When fault intensity constant value is defined but the range is outside the limit (0-1),

runner.registerError("Damper position must be between 0 and 1 and it is now #{damper\_pos}!")

### Information

* Works with,
  + Controller:OutdoorAir.

### Code Outline

* Define arguments (economizer where the fault occurs, schedule of fault presence, damper position under faulted condition).
* Check whether fault intensity value is valid between 0-1.
* Find the economizer where the fault occurs (check whether economizer option is enabled) and impose fault to the economizer.
  + If user defined fault presence schedule is available, define fault schedule according to this fault presence schedule and damper stuck position.
    - Create default day faulted damper schedule.
    - Create overriding ScheduleRules.
    - Create summer design day faulted damper schedule.
    - Create winter design day faulted damper schedule.
    - Apply faulted damper schedule to selected economizer’s min & max. outdoor air fraction fields.
  + Else, create a schedule based on the damper stuck position value and apply it to selected economizer’s min. & max. outdoor air fraction fields.

### Tests

* Test invalid user argument values to make sure measure fails gracefully
* Test fault implementation with and without the fault presence schedule