Jeff's heating control algorithm

Author Jeff Thompson

Contact Domoticz forum - @jefft

Github repo github.com/jefft4/Heating-control-algorithm

Version 2.1; October 2020

User Guide

Features

- Handles any heat source and heater type that can be represented in domoticz as the required simple devices
- Works with boilers / heat sources where the temperature can be controlled and those where it's fixed; uses both temperature modulation and time modulation
- Works with any valve type (best used with TRVs to balance heat between rooms within zones)
- Predicts heating required to meet next setpoint (e.g., warm rooms ready for your arrival home, no cold bedrooms when you wake up), allowing for heat loss to outdoors
- Any number of zones; handles both hot water and heating
- Allows for heat source warm-up time
- Zones configurable - indoor and outdoor temperature sensors, setpoint, two valves (master and local), heating and heat loss rates, preferred and maximum pre-heating periods
- Heat source configurable – warm-up rate, maximum and minimum temperatures, minimum on time

Setting up

To get things up and working, you need to copy the code into a new event in Domoticz, set up the triggers, and then create a heat source, one or more heating zones and their associated sensors and valves.

The code

Clone the code from the Github repo using git or by downloading the Zip file. There's one file of code. Create a new event in Domoticz – type dzVents/minimal – delete all the pre-populated code, then copy and paste all of the heating algo code into the event and save it. I recommend setting it 'OFF' at the top until you've set up the devices and variables that drive it.

There are some things that you need to change in the algorithm dzVents code to make it run in your system:

- Near the end of the file, in the main event code block, are the dzVents triggers for the event. Set the 'timer' clause to suit you; this time should match the 'cycleTime' value in the global user variable (see below). Normally, 15 minutes is a good choice.
- Set the 'devices' list to include all your heating zone setpoint devices. This is so that manual and scheduled changes to the set heating temperatures cause the algo to run and re-assess the heating situation. If you omit these, the algo won't react to setpoint changes until its next scheduled 15 minute run. It may be wise not to set up a lot of these initially until you've got things working.
- Look at the code blocks for the boiler master switch and the service switch. If you want these, either create two dummy switches of those names, or change them to the names of your own equivalent switches. If you don't want this functionality, remove the two if…end blocks.
- Decide whether you want the full debug level logging on; in a live system, I wouldn't, but it is helpful for seeing the calculations and decisions that the algo makes while you're setting up. Comment (--) or uncomment the relevant 'DEBUG_INFO' line in function lognote, right at the top of the code.

Once you've created the event and adapted the triggers to suit your system, you'll need to set up some devices and user variables to define and drive the system. Read on!

Heat source

There's just one of these. It's defined by a single user variable.

Devices:

- A setpoint device for its output temperature; if your source isn't controllable, the device can be a dummy setpoint but it still needs to be there
- A temperature sensor for its output ('flow') temperature and one for its 'return' temperature. If there's only one temperature (e.g. electric heating) then these can be one device. If you don't have measurement of one of these temperatures, create a dummy device and fix its temperature at a sensible value. *More on this in the Questions section.*

User variable: 'Heating control global'

- JSON string, enclosed in {}

"cycleTime"	Integer, minutes	Normal period for reviewing heating situation ** Must match the dzVents time trigger clause!
"heaterMinTemp"	Integer, degC	Minimum output temperature of the heat source
"heaterMaxTemp"	Integer, degC	Maximum output temperature of the heat source
"heaterMinOnTime"	Integer, minutes	Minimum on-time of the heat source (to prevent short cycling/damage)
"idxFlowTemp"	Integer	Domoticz idx of the flow temperature sensor
"idxReturnTemp"	Integer	Domoticz idx of the return temperature sensor
"warmUpRate"	Integer, degC/min	How fast the heat source output warms up
"flowSet"	Integer	Domoticz idx of the setpoint device for the heat source's output temperature

Zones

A heating zone is a distinct part of the building for which you want to control the temperature and ideally, to which you can control the supply of heat independently of other zones.
Zones are defined by a user variable each. The name must begin 'Heating zone '; the rest of the name is purely descriptive.

Devices:

- A setpoint device (which you can use for scheduled temperatures and manual override). This can be a smart TRV's setpoint – the algorithm has a lockout to prevent loops when it changes the setpoint.
- A temperature sensor device. This can be a smart TRV's sensor device.
- Associated valve device(s) – see 'Valves' below.

User variable: 'Heating zone X'

- JSON string, enclosed in {}

"sensor"	Integer	Domoticz idx of the temperature sensor for the zone
"age"	Integer, minutes	Maximum age of a sensor reading to be considered valid
"setpoint"	Integer	Domoticz idx of the zone's setpoint device
"valve"	String	Local zone valve ID for this zone – see Valves below
"mvalve"	String	Master valve ID for this zone – see Valves below
"incRate"	Float	1000 * degC rise/minute/degC delta, rate of heat input to the zone (delta = source temp – room temp)
"decRate"	Float	1000 * degC rise/minute/degC delta, rate of heat loss from the zone (delta = room temp – outdoor temp)
"outSensor"	Integer	ID of the zone's outdoor temperature sensor – see Sensors below
"prefPHT"	Integer, minutes	Preferred pre-heating time ahead of setpoint rise
"maxPHT"	Integer, minutes	Maximum pre-heating time ahead of setpoint rise
"minDT"	Integer, degC	Minimum useful difference between source and room temperatures
"hyst"	Float, degC	Hysteresis – how far we let the temperature vary from the setpoint before we do something about it

Either valve or mvalve must be valid; the other is then optional.

Valves

A valve controls the supply of heat to a zone. The algorithm treats all valves as being either on or off, but will work with valves that need other control approaches like % open or heat mode.

Valves are defined by a user variable which describes the actions to turn them on and off. Valve ID 'Y' as used in a zone will be defined by user variable 'Heating valve Y'.

Devices:

- Any of:
    - On/off switch
    - Percentage switch (dimmer)
    - Thermostat mode device
    - Thermostat setpoint device

User variable 'Heating valve Y':

- JSON string, enclosed in {}

"idxOnOff"	Integer	Domoticz idx of zone on/off switch
"idxPercent"	Integer	Domoticz idx of zone percentage switch (typically valve open %)
"idxMode"	Integer	Domoticz idx of zone thermostat mode device
"strModeOn"	String	String value to set the mode device to heat
"strModeOff"	String	String value to set the mode device off (or to Eco mode if preferred)
"idxSetpoint"	Integer	Domoticz idx of zone setpoint device

- At least one of the four idx values must be valid. If it's idxMode then the two strings are also required. All devices present will be processed each time the algorithm changes the valve state, in the order listed.

Sensors

Zone temperature sensors are simple domoticz temperature (or TH/THB) devices – anything that will return a result when dzVents asks for device(x).temperature. These are defined in the zone by their Domoticz idx value.

Zone outdoor temperature sensors can include future predictions of the temperature, taken from a weather forecast API or other source. These are defined in the zone by a string ID – e.g. 'OUT1', which references the user variable 'Heating sensor OUT1'.

Devices

- For zone (indoor) sensor, any temperature device.
- For outdoor sensor, any temperature device for current temperature plus 4 temperature devices for the next 4 hourly predicted temperatures.
- If prediction is not available or not required, then the predicted temperature sensors can be the same device as the current temperature sensor.

User variable 'Heating sensor OUT1':

- JSON string, enclosed in {}

"idxNow"	Integer	Domoticz idx of the current temperature sensor

“idx1”	Integer	Domoticz idx of the temperature sensor for 1 hour in the future
“idx2”	Integer	Domoticz idx of the temperature sensor for 2 hours in the future
“idx3”	Integer	Domoticz idx of the temperature sensor for 3 hours in the future
“idx4”	Integer	Domoticz idx of the temperature sensor for 4 hours in the future

Tuning the algorithm’s behaviour

The algorithm is driven by a number of parameters defined in the user variables:

- incRate and decRate for each zone define how quickly that zone heats and cools
- warmUpRate (global) defines how long it takes the heat source to warm up
- minDT, hyst and the other heater parameters also have some effect but are less system-specific.

Setting the heating rates

To calculate the incRate and decRate, use real data. Make an initial estimate, then observe your system’s behaviour over a period of time and use what you observe to fine-tune it.

For incRate, try to look at situations where most/all zones are heating constantly for a period of time (the algorithm assumes that your heat source is big enough to heat all zones together). Divide the degC temperature increase for a zone by the time taken to get that increase (in minutes) and then divide by the degC average difference between the heat source temperature and the zone temperature. Finally, multiply by 1000 – we do this just to make it more friendly in the user variable and because domoticz user variables are limited to a rather mean 200 characters!

For decRate, it’s a similar approach. The best period to sample is probably just after your heating switches off for the night, through until the lowest zone temperature is reached or the heating switches on again. Same calculation as for incRate, but use the decrease in zone temperature rather than the increase (the result needs to be a positive number).

If you want a starter for ten; my heating rates vary between 0.3 for a room with a slightly undersized radiator and 1.2 for the hot water cylinder. Upstairs increase rates are higher, thanks to the laws of thermodynamics – downstairs zones heat upstairs to an extent whether I like it or not. My decrease rates range from 0.44 for a room with an uninsulated north wall and 8sq.m of glass to 0.2 for the main upstairs zone.

Other tweaks

If you find that your rooms are getting a little too cold before the heating kicks in, reduce the hysteresis value. Conversely, if you find that your heating is running a lot for short periods and possibly over-heating you then letting you cool off (it has to obey minimum on periods and temperatures), then try increasing the hysteresis.

If you want a suggestion, I’d set hysteresis to 0.5 for room zones and perhaps 3 or even 5 for stored hot water. My minDT is 10 degrees – that is, radiators need to be 10degC above the room temperature before they have much effect.

Questions

How does the algorithm play nicely with TRVs?

The algorithm will work with both dumb and smart TRVs, and a mixture of the two. I personally have a handful of smart TRVs in tricky rooms and dumb ones everywhere else.

Through my testing, I've found that 'smart' TRVs are rarely all that smart. They have a schedule, a temperature sensor and a valve. The logic connecting those three things doesn't generally attempt prediction, allow for warm-up times, read the weather or anything else; those advanced calculations are the job of central controllers in systems like Tado and Bosch.

If you set a temperature schedule on a smart TRV and then use it with this algorithm, it may 'fight' with the algorithm during pre-heating periods as the valve still thinks that the temperature should be low, while the algorithm is trying to warm up ready for the coming increase. If that happens, use a dummy Domoticz setpoint for the schedule instead.

Open window detection can be enabled – it causes the valves either to close or to reduce the set temperature. If the valve closes without dropping the set temperature, the algo won't know and so will continue to try to heat (possibly more so as the temperature will have dropped). If you want to be smarter and more robust with open door/window detection, consider having some custom code that's triggered by door & window sensors and reduces zone setpoints temporarily… I have some that I'll publish as a starter.

TRVs can be opened and closed by one or more of several commands; that's why we have the four options in the valve definition. For example, my Eurotronic Spirit Z-Wave TRVs can be controlled by setting the mode + setpoint, or by setting the mode to 'Manufacturer specific' and then setting the valve %.

The algorithm does not try to make use of **partial (%) valve opening**. That might be something I'll consider in a future enhancement. It's really only useful if you can't control the heat source temperature, when partially closing the valve is a potential substitute… however, it's only a reasonable substitute if all the heaters in all the zones have valves that can be controlled in that way – otherwise you'll get full heat in those with dumb, or no, valves which could ruin all the hard work in the smarter zones. Modulating the heat by turning the heat source on for only part of each cycle works pretty well; the algo will do that automatically.

Dumb, mechanical TRVs – and some smart ones - may gradually close as they get close to the set temperature. That may slow the heating a bit as the zone gets close to target, possibly leading to the zone slightly undershooting the target temperature or achieving it a little late. If that happens too much, you might consider setting the valve a little higher than your highest target temperature. Smart TRVs should be told what to do by the algorithm; if they are deciding to close as they approach the target and causing an undershoot then check whether you can control them by a different approach (e.g., having the algorithm set valve % rather than, or in addition to, setting the mode).

What do I use for the heat source set temperatures and sensors, if I don't have / can't control those?

If you don't have control of the max and min source temperatures (the flow temperature of most boilers is set by a knob on the front panel, unless you have an OpenTherm or EMS gateway device

connecting them to Domoticz), then find out what the supply temperature is and set both max and min to that value. Let it warm up fully before measuring temperature and measure directly on the copper flow pipe from the boiler/source to the heaters.

If you can't feed Domoticz with the flow and return temperatures, then set up a dummy temperature sensor device and force its value to the measured flow temperature. The return temperature isn't currently used in the algorithm, so just point it to the same device. I do have an idea to use the return temperature in the future to improve the warm-up calculation; if I get around to that, it'll come with new advice.

I set my Bosch EMS boiler's flow temperature but a few minutes later, it resets to the value on the control panel!

Yes, that's right. Welcome to the wonders of EMS! Unfortunately, EMS boilers seem to revert to the panel setting if they've not had a contrary command in the last few minutes. I get around this in my system with an event that runs every 2 minutes and re-asserts my view of what the flow temperature ought to be. Happy to share that little piece of code if you need it, just ask me!

How do I handle water heating?

If your heat source also heats your hot water, as long as there are devices that the algorithm can treat as the valve and sensor for the hot water 'zone', then it's just another zone; one with much higher target temperatures! I define my hot water with the indoor temperature from a sensor on the cylinder and the outdoor temperature from the garage (that's where the cylinder is) – all five outdoor idx's point to the same sensor).

When do I need to use the local valve and master valve?

If all your system needs to do to heat a zone is to activate one valve/device, then use the 'valve' (local valve) in the zone definition. If it needs to activate more than one device, you'll need the master and the local. For example, my system is an old-style 'S-plan' system with a master valve each for heating and hot water, plus I have some TRVs in specific zones. So, I have the 'mvalve' for all my heating zones set to the S-plan heating valve and the 'valve' set to the local TRV where there is one. Where there isn't a local TRV, I just omit the local valve element from the zone definition.

There is no difference in how the two are handled; the naming is just intended to give a clue as to their use.

What if I don't have some of the sensors?

A zone must have its zone (indoor) temperature sensor and outdoor sensor. If you don't have an outdoor sensor, then set up a weather forecast API in Domoticz and take the current & future temperatures from that.

Do sensors and/or valves need to be unique to a zone?

No.

Multiple zones often share something like a S-plan heating master valve.

The outdoor temperature sensor is likely to be shared by most/all zones as there's only one outdoors (though if you want to be really clever and account for solar heating as the sun moves around, you'll probably want more than one).

Indoor sensors are generally unique to one zone, although setting up multiple zones with a shared indoor sensor could be a solution if you have, for example, 4 bedrooms each with a separate valve but no local sensors, just one in the hallway.

Can the algorithm handle more than one heat source?

Not in a single system. You'd have to work out a way to run more than one copy of the code and use a different user variable for the global data – or adapt the code so that each zone specifies its heat source and there's a global data chunk for each source. Probably not that difficult; if you get it working, make a request to put it into the git repo for everyone to use!

Troubleshooting

I'm sure this section will grow as people use the algorithm!

For now, I'll say these key things:

(1) Check your definitions carefully. Don't add ten of them all at once; add enough to drive the source and one zone, test, then add more if all's well.
(2) If the behaviour isn't as expected, edit the code of the function 'logNote' to uncomment the line 'DEBUG_INFO = true' and to comment (-- at the front) 'DEBUG_INFO = false'. This will cause the algorithm to write a ton of detail to the domoticz log, from which you should be able to see its thinking and what values were used.
NB! The web UI's log page sometimes doesn't keep up with the volume of records written by the algorithm in a short time; look at the domoticz.log file directly to be sure you see everything.
(3) Check the log for warnings / errors.
(4) Ask on the forum. Other people will often be the best source of ideas and experience. I'll also try to monitor the thread and respond as quickly as I can.

Example systems – how they could work

1. My system: Wet CH & HW, S-plan plumbing, Worcester-Bosch Greenstar gas system boiler with temperature control and sensing via a EMS-ESP module. Mostly dumb TRVs, some smart ones in the tricky rooms. Also a smart on/off control for the kitchen heaters.

   Heat source (Heating control global): {"cycleTime": 15, "heaterMinTemp": 35, "heaterMaxTemp": 75, "heaterMinOntime": 5, "idxFlowTemp": 1433, "idxReturnTemp": 1366, "warmUpRate": 4}

   Zones: Downstairs general ('Hall'), Upstairs general ('Landing'), Kitchen, Lounge, Study, Bedroom 3, Bedroom 1, Hot water.

   Indoor sensors: Hall, Landing, Kitchen, Lounge, Study, Bedroom 3, Bedroom 1, Hot water.

   Outdoor sensors: single real sensor for current temperature, weather forecast API provides 4 x forecast temperatures. Hot water cylinder uses garage temperature as current and future 'outdoor'.

   Valves: Master HW, Master CH – the S-plan valves. Zone local valves in Study, Lounge, Kitchen. Other zones have no local valve defined.

2. Underfloor heating; each zone has its own sensor and valve

Heat source: {"cycleTime": 15, "heaterMinTemp": 30, "heaterMaxTemp": 50, "heaterMinOntime": 5, "idxFlowTemp": 1002, "idxReturnTemp": 1001, "warmUpRate": 2}

Zones: one per physical heating zone
Indoor sensors: one per physical heating zone
Outdoor sensors: one real, 4 from weather forecast
Valves: each zone has master valve set to the boiler control (on/off) and local valve set to the relevant physical zone valve

3. Wet CH & HW, smart TRV in every room, no real outdoor sensor
   Heat source: same as example 1
   Zones: one per TRV/room
   Indoor sensors: TRV sensor devices
   Outdoor sensors: all from forecast API
   Valves: TRV valve devices