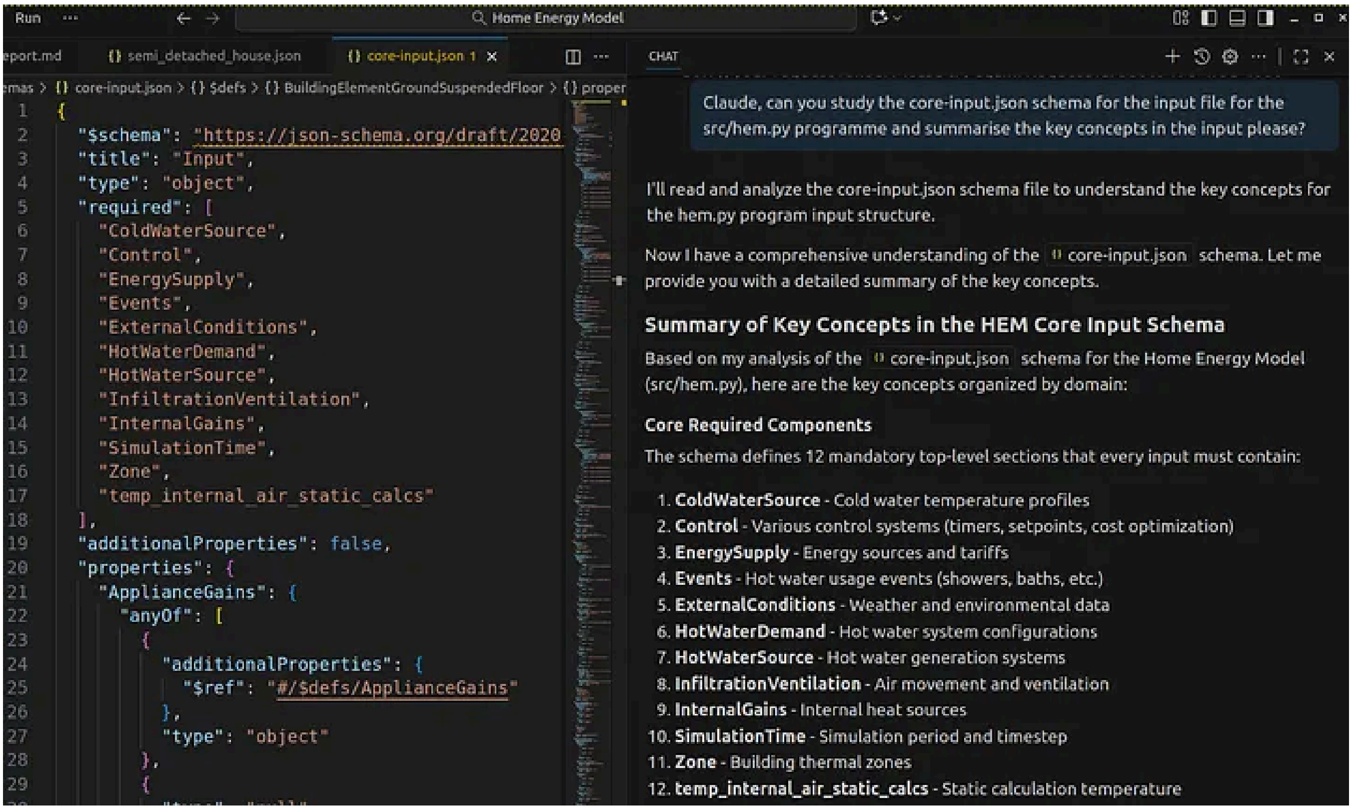
Productivity Increase using AI models — Creating a Home Energy Model

9 min read · 3 days ago

Peter Wurmsdobler

Over the past months I have been using Microsoft Copilot for various projects both in a professionally and private context. In this post I would like to share my experience using Copilot in agent mode in order to analyse the reference implementation of the UK government's Home Energy Model (HEM), as well as to create an input configuration file for a specific property, namely our 1930ies semi-detached house.



VSCode showing the Home Energy Model schema as well as the conversation with Claude.

Home Energy Model

The Home Energy Model (HEM) is the UK Government's proposed National Calculation Methodology for assessing the energy performance of dwellings. Sometimes referred to as SAP-11, it is supposed to replace the Standard Assessment Procedure (SAP), now at SAP 10.2; all are used to produce an Energy Performance Certificate (EPC) required for selling or renting out a property in the UK. What is most interesting, however, is that the reference implementation is available under an MIT license.

The first step was of course to `git clone` the repository into a local folder on my computer (Ubuntu 22.04.2 LTS), open the project in VSCode, study the `REAMDE.md`, follow its instructions to create a virtual python environment — as is common practice — , download a weather file from Climate.OneBuilding.Org (Cambridge in my case), and finally run the simplest of all calculations using the demo input configuration `test/demo_files/core/demo.json`:

```
python src/hem.py test/demo_files/core/demo.json \
    --epw-file $HOME/Downloads/GBR_ENG_Cambridge.AP.035715_TMYx.2004-2018.epw
```

After a successful run, the folder `test/demo_files/core/demo__results` contains some files that looked sensible, e.g. the `demo__results__static.csv` with an entry for the *"Heat transfer coefficient,W / K,173.915441"*; of course this value is not meaningful as the input configuration does not represent a real property. At this point I am confident to have at my disposal a working HEM.

Iterations

My objective was to create an input configuration that is representative for our 1930ies semi-detached house, run a simulation using the Cambridge weather data, and produce the standard output which should include the property's heat transfer coefficient as well as estimated annual energy consumption. The source code root folder shows `src` , `schema` and `test` folders, but no sign of documentation about a process or guideline to create the input configuration for a specific property, not even concept documents or object model diagrams. Since I do not fancy to reverse-engineer the input schema, I used Copilot, Claude Sonnet 4 or short *Claude*, for an initial analysis:

```
> Can you study the core-input.json schema for the input file for the src/hem.p
```

Claude responds by rephrasing the question, by reporting that it does analyse the schema as well as the python program, then produces a report about its findings (showing the first few lines):

```
GitHub Copilot: Now I have a comprehensive understanding of the core-input.json

## Summary of Key Concepts in the HEM Core Input Schema

Based on my analysis of the core-input.json schema for the Home Energy Model (s

### **Core Required Components**
The schema defines 12 mandatory top-level sections that every input must contai

1. **ColdWaterSource** - Cold water temperature profiles
2. **Control** - Various control systems (timers, setpoints, cost optimization)
3. **EnergySupply** - Energy sources and tariffs
4. **Events** - Hot water usage events (showers, baths, etc.)
5. **ExternalConditions** - Weather and environmental data
6. **HotWaterDemand** - Hot water system configurations
7. **HotWaterSource** - Hot water generation systems
8. **InfiltrationVentilation** - Air movement and ventilation
9. **InternalGains** - Internal heat sources
10. **SimulationTime** - Simulation period and timestep
11. **Zone** - Building thermal zones
12. **temp_internal_air_static_calcs** - Static calculation temperature
```

Claude elaborates further on Building Thermal Model, Zones, Building Elements, Thermal Bridging, HVAC Controls, External Conditions and Internal Gains, etc. At least I get an idea of the scope of the model and what it is capable of representing. So I continue:

```
> How would I go about creating an input json file for a semi-detached two-stor
```

Claude obliges by saying that it will help me creating an input `json` for a semi-detached two-storey, three bedroom house with central heating and a gas boiler.

Then Claude continues analysing the structure of all demo files which it discovered on its own; after a few minutes it produces:

```
## Summary

I've created a comprehensive guide for building an HEM input JSON file for your

### **What I've Provided:**

1. **Detailed Guide** (semi_detached_house_guide.md) - Explains all the key con
2. **Working Example** (semi_detached_house_example.json) - Complete, functiona
3. **Step-by-Step Workflow** (step_by_step_workflow.md) - Systematic process fo

### **Key Steps to Follow:**

1. **Start with the example file** - Copy semi_detached_house_example.json and
2. **Work incrementally** - Make small changes and test frequently
3. **Begin with short simulations** - Use 168 hours (1 week) initially, then ex
4. **Customize systematically**:
   - Building geometry and dimensions
   - HVAC system specifications
   - Control schedules
   - Occupancy patterns
```

I study the generated guides and example file, but I want to make sure that a simulation run actually would work:

```
> Can you make sure that the example input file works with:
python src/hem.py semi_detached_house_example.json  --epw-file $HOME/Downloads/
```

It realises that the execution fails, but after a several iterations (thanks to the agent mode) using a shell to run the python scripts and analysing the error messages, Claude fixes the example input file, and produces a summary (abbreviated):

```
## Successfully Created Working HEM Input File

Your **semi_detached_house_example.json** file now runs successfully with the H

### Key Results from the Simulation:
```

```
**Energy Consumption Summary:**
- **Total Gas Consumption**: 388.24 kWh (primarily for heating and hot water)
- **Total Electricity Consumption**: 5.81 kWh (boiler auxiliary power)
- **Space Heat Demand**: 2.68 kWh/m² (very efficient for a 168-hour period)
- **Peak Electricity Demand**: 0.077 kWh (during evening heating period)
```

At this point I have a working base model and commit the local changes, then follow the advice to gather the following data:

```
## Phase 1: Planning and Data Collection

Create a checklist of required data:

**Building Geometry:**
- [ ] Floor area (each floor separately)
- [ ] Room heights
- [ ] External wall dimensions
- [ ] Window sizes and positions
- [ ] Roof area and pitch
- [ ] Party wall area (shared with neighbor)

**Construction Details:**
- [ ] Wall construction type and U-value
- [ ] Window type (single/double/triple glazed)
- [ ] Floor construction (slab/suspended)
- [ ] Roof insulation level
- [ ] Airtightness test results (if available)

**HVAC Systems:**
- [ ] Boiler make, model, and rated power
- [ ] Hot water cylinder size
- [ ] Radiator types and sizes
- [ ] Thermostat/programmer details

**Occupancy:**
- [ ] Number of occupants
- [ ] Typical daily schedules
- [ ] Hot water usage patterns
```

Once I have all required information gathered in a file `specification.txt` more or less in a `yaml` kind of format I ask:

```
> Now that we have an initial version, can you adjust that version to meet the
python src/hem.py semi_detached_house_example.json --epw-file $HOME/Downloads/G
```

and after thinking a bit Claude reports:

```
Now I understand the detailed specifications. This is a much more complex build

**Key differences from current model:**
1. **Size**: 5.8m × 9.0m footprint (52.2m² ground floor + extension)
2. **Detailed room layout**: 10 specific rooms with individual dimensions
3. **Boiler**: 26kW (not 24kW)
4. **Hot water tank**: 120L (not 180L)
5. **Multiple zones**: Each room has specific dimensions, orientations, and the
6. **Mixed floor types**: Suspended wood floors + concrete slab in extension
7. **Complex roof geometry**: Mix of flat ceilings and slanted roofs at 45° pit
8. **Detailed building fabric**: Specific U-values for walls, windows, doors
```

Claude goes off and creates a 5-step plan to make the required changes, then runs the simulation, discovers some `json` validation or runtime errors, fixes them until a final model is generated, and summarises:

```
## Summary of Completed Work

### ✅ Major Model Restructuring
- **Converted from 2-zone to 10-room model** to match the detailed specificatio
- **Created individual zones** for each room: entrance hall, front room, family
- **Individual heating systems** with thermostatic radiator valve (TRV) control

### ✅ Detailed Room-by-Room Implementation
- **Accurate dimensions and volumes** per specifications (e.g., entrance: 1.5×4
- **Proper building elements** with correct orientations, U-values, and areas f
- **Correct floor types**: suspended floors for ground floor rooms, slab floor
- **Proper external wall configurations** with specified U-values (0.37-2.05 W/

### ✅ Individual Heating Control Systems
- **8 separate heating systems** for heated rooms with individual radiators and
- **Differentiated heating schedules**:
  - Standard rooms: 16-21°C
  - Bathroom: 16-22°C (higher temperature)
  - Bedrooms: 15-18°C (lower temperature for sleeping comfort)
- **Properly sized flow rates** based on room sizes and heat requirements
```

```
### ✅ Building Physics Accuracy
- **Fixed suspended floor elements** by adding required fields (`height_upper_s
- **Accurate thermal bridging values** and heat capacities
- **Proper window and door specifications** with correct U-values and orientati
```

Claude also discovered that there is a bug in the HEM as two unheated zones create a conflict in an internal dictionary. The workaround is a simple dummy heating with zero output for those rooms. After resolving a few other snags, I wanted to extend the simulation to a full year:

```
Is there a provision in the events section to also use a schedule and then repe
And in places where there is now a daily schedule that is repeated in the main
Since we have all working for one week with the daily pattern repeated 7 times,
```

Claude upscaled the model identifying all the elements that need changing and produced an input configuration for a whole year making use of nested, named patterns:

```
## ✅ **Yearly Model Successfully Completed!**

**Summary of achievements:**

1. **✅ Extended to Full Year**: Simulation time increased from 168 hours (1 we

2. **✅ Updated All Schedules**: All internal gains and control schedules now r

3. **✅ TRV Controls for Full Year**: All thermostatic radiator valve controls

4. **✅ Yearly Shower Events**: Generated and integrated 365 daily shower event

5. **✅ EPW Weather Integration**: Model correctly uses the Cambridge EPW weath

**Key Results from Full Year Simulation:**
- **Annual Space Heat Demand**: 252.6 kWh/m²
- **Total Gas Consumption**: 14,755 kWh/year
- **Total Electricity**: 637 kWh/year
- **System Efficiency**: 0.86-0.90 CoP across different zones
- **Peak Electricity Demand**: 0.12 kW (February, 8 AM)
```

These values are already in the right ball park. For me it takes some more iterations to adjust parameters which I gradually performed by first asking questions about the schema such as:

- How would a flat ceiling be modelled that has an unheated roof space above?
- How does the schema configure an internal wall between two adjacent rooms?
- It is surprising that HEM does not need to know which other room a given room is adjacent to as the energy being passed is dependent on the temperature difference between both rooms.

Claude gave very good answers to all questions and helped me to understand some underlying concepts in order to arrive at a decent model.

Conclusion

The presented workflow is a distilled version of a longer conversation, and in a certain way a sanitised version, but the process is clear: ask Claude to analyse the schema (or source code), then to build a simple model and gradually improve it by adding complexity while always making sure that the calculation run succeeds. Copilot (Claude in my case) in agent model will use any output errors to detect missing input quantities or inconsistencies and fix them, or if it cannot be fixed, indicate where the issues are or even discover a bug in the source code.

Beyond the initial objective, the process of building and enhancing a input file allows learning the schema and implicitly create a model that is suitable for my case; there is a myriad of possibilities of creating a thermal model for a semi-detached house. What is important is the gauge the output, i.e. verify that the model converges and that the predicted thermal losses or energy spent correlate to the recorded data, e.g. the annual energy bill; the desired outcome drives the cooperation with Claude.

References

- Energy Performance Certificates (EPC), Department for Energy Security and Net Zero
- Standard Assessment Procedure, Department for Energy Security and Net Zero, Published 22 January 2013 Last updated 14 May 2025.
- SAP-10.2 — Standard Assessment Procedure for Energy Rating of Dwellings, Building Research Establishment (BRE), 2021.

- RdSAP-10 — Reduced data Standard Assessment Procedure for Energy Rating of Dwellings, Building Research Establishment (BRE), 2025.
- Home Energy Model: technical documentation, Department for Energy Security and Net Zero Published 13 December 2023.
- Home Energy Model Reference Source Code Repository.

Microsoft Copilot | Ai Tools | Anthropic Claude | Home Energy Management | Building Energy Modeling

Follow

Written by Peter Wurmsdobler

317 followers · 21 following

Interested in sustainable mobility, renewable energy and regenerative agriculture as well as music and audio.

No responses yet

Bgerby

What are your thoughts?

More from Peter Wurmsdobler

Peter Wurmsdobler

Productivity in Software Development using AI models — A Pilot Project

My experience of using Microsoft Copilot on a small pilot project led me to appreciate a new way of software development.

Oct 14 79 3

Peter Wurmsdobler

Improving the Thermal Performance of UK 1930s Semi Detached Houses

Sharing the experience of improving the thermal performance of a UK 1930s semi detached house and suggestions for what could be done…

Jan 7, 2021 68 6

Peter Wurmsdobler

Building a Wooden Electric Go-Kart

What started as a soapbox ended up as an electric go-kart; apart from all moving parts, made from timber planks during the recent…

May 8, 2021 67 1

Peter Wurmsdobler

Possible Personal Contributions towards a Sustainable World

Explore my options given the state of the world and my abilities.

May 5 238 18

See all from Peter Wurmsdobler

Recommended from Medium

Ignacio de Gregorio

Continual AI Learning, Finally Solved?

A Smart Trick for a Hard Problem

3d ago 602 10

In Entrepreneurship Handbook by Joe Procopio

AI Porn Is Here—and We're All Pretending It'll Be Fine

Sam Altman has decided that five years is enough of a head start for the porn companies

4d ago 1.2K 50

In Towards AI by Teja Kusireddy

We Spent $47,000 Running AI Agents in Production. Here's What Nobody Tells You About A2A and MCP.

Multi-agent systems are the future. Agent-to-Agent (A2A) communication and Anthropic's Model Context Protocol (MCP) are revolutionary. But...

Oct 16 2.5K 80

In Coding Nexus by Code Coup

Claude Desktop Might Be the Most Useful Free Tool You'll Install This Year

I didn't expect much when I first saw the announcement for Claude Desktop. Another AI wrapper, I thought. Maybe with a shiny UI.

Oct 23 551 23

In Level Up Coding by Jacob Bartlett

The Terrible Technical Architecture of my First Startup

A system that could only be built with the confidence of a 25-year-old Deloitte graduate with an AWS certification

4d ago 267 7

In UX Collective by Neel Dozome

After the AWS outage

Can we trust Big Tech monopolies in essential infrastructure?

4d ago 323 2

See more recommendations