Digital Representation of Heating Systems for Fault Detection Purposes

Matthias Georgii[1], Christoph Schmelzer[1], Christian Sauer[1], Janybek Orozaliev[1], and Klaus Vajen[1]

[1] University of Kassel, Institute of Thermal Engineering, Kassel (Germany)

Abstract

A concept for a graph based digital representation of heating systems is presented. It explicitly defines various relations between the components and data points. Thus, it allows a semantic description of the heating system and available data points which can be used for automated fault detection. Implementation of analysis algorithms can get complex considering the multitudes of options regarding different system configurations and measurement equipment. The semantic description enables a computer to make automatic conclusions, and thus can reduce this effort. Simultaneously, the concept also may reduce the effort to find and apply analysis algorithms for a specific system, since it allows for automated comparison between needed and available components along with their relations.

Keywords: function control, fault detection and diagnosis, FDD, Digital Twin

1. Introduction

Heating systems are getting more complex to achieve high efficiencies and high fractions of renewable energy. At the same time, they are often individually planned and assembled at each site, with individual parameterization of controller settings. This may lead to faults in planning, installation, and operation phases. Automated fault detection and diagnosis (FDD) can help to ensure the intended behaviour and efficiency of the system. While currently many sensor and signal data can be logged, the automated analysis of the data is less common. Often, data is just visualized and interpreted by an expert. This might have two causes.

Firstly, there is little or no information about validated detection algorithms publicly available. For instance, VDI 2169 (VDI, 2012) mainly describes rather rough concepts of detection ways than directly applicable algorithms. However, when implemented in the simple form, they will rise lots of false alarms, making them practically unusable.

Secondly, it is challenging to automatically select applicable detection algorithms and link available data to their inputs. These tasks of selecting and linking require meta information, which often is not stored at all, or just in an inconsistent, heterogenous way, which impedes easy access by machines (see fig. 1). In single monitoring projects, an expert uses his meta information knowledge to select the algorithms that should be executed, and to manually assign suitable data to their inputs. This approach would lead to high effort for greater numbers of monitored systems. For an automated analysis of many different systems, each heating system has to be represented digitally in a way that allows the automatic application of detection algorithms. One possibility is to use predefined blocks of components, as done in the Methodiqa project (Ohnewein et al., 2016). This works well when small parts like components or a group of nearby components are considered, and a limited number of detection paths. However, the block representation approach might get complex, if interactions of larger parts of the system or the whole system are to be considered, and more different sets of available data points and correspondingly different ways of deriving information have to be checked.

We therefore suggest a graph representation which explicitly states relations between the components or subsystems of the system and offers more flexibility.

doi:10.18086/eurosun.2022.16.04 Available at http://proceedings.ises.org

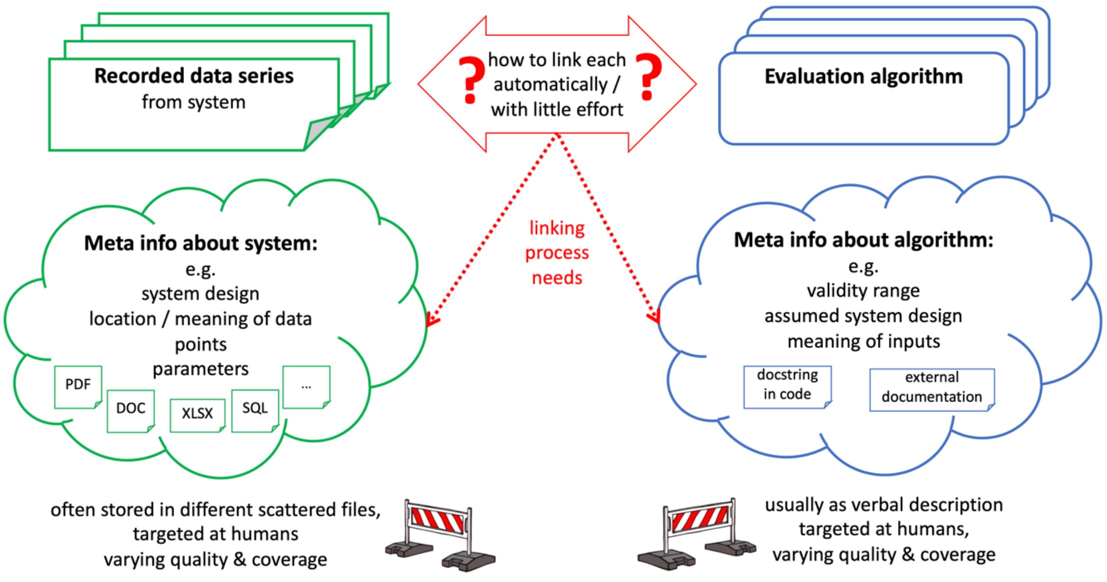
Recorded data series
from system
how to link each automatically / with little effort
Evaluation algorithm
linking process needs
Meta info about system:
e.g.
system design
location / meaning of data points
parameters
PDF
DOC
XLSX
SQL
...
Meta info about algorithm:
e.g.
validity range
assumed system design
meaning of inputs
docstring in code
external documentation
often stored in different scattered files, targeted at humans varying quality & coverage
usually as verbal description targeted at humans, varying quality & coverage

Fig. 1: Selecting applicable evaluation algorithms and linking them to recorded data requires meta information which often is not available in a consistent, machine-readable manner.

2. Graph representation

Depending on the intention, different views on a heating system are important.

- Geometry and Topology: distances and dimensions of subsystems and components, or their spatial adjacency or inclusion, e.g. lengths of pipes
- Function: what is the function of which subsystem, functional grouping of subsystems, what are intended interactions, e.g. supply pipe delivering heat from boiler to storage
- Control: which interactions between subsystems are actively controlled by which controller, and which of their signals are used for which control task, e.g. storage bottom temperature used as input for operating the solar circuit via a temperature hysteresis control
- Recorded data structure: identification, availability, and logical grouping of data points, in order to find the meaning of each time series of data
- Abstraction level of system components: for instance, an air-to-water heat pump might also be regarded as a general heat pump or heat generator or energy converter

Thus, those aspects should preferably be included in the graph representation. For FDD, geometry is the least important aspect and only needed in some specific evaluations, which is why it is omitted here. The function of a subsystems can also be considered in many respects. Since the purpose of a heating system is to generate heat and provide it to different consumers, it is useful to describe the function with respect to both the energy flow and the fluid flow that may transport the energy.

2.1. Energy flow graph

The energy flow graph describes the energy flows from heat generators to heat consumers. It already contains the basic structure and main components of a heating system. An energy flow graph that just contains heat generators, storages and sinks is illustrated in fig. 2. However, since a component, e.g. a storage, can have several ports at which energy is transferred, those ports have to be included as nodes as well in order to have nodes that uniquely identify a specific energy flow. An example of such an extended energy flow diagram is shown in fig. 3. It additionally contains the solar heat exchanger (which does not alter the energy flow in first approximation) and “logical” components (“XORDistributor”, “Adder”) that explicitly distribute or merge energy flows, ensuring that there is only one incoming and one outgoing edge at each port node. This prevents ambiguities how to interpret several incoming/outgoing edges at a node. Of course, each component still needs an individual node representing the component, in order to describe, to which component an energy port belongs. The component node has the type of the component. Such a component node also indicates that energy flow (and/or state variables of associated medium flow) is significantly altered by conversion, merging, distributing, etc. For instance, a SolarThermalCollectorField node will convert irradiance into thermal energy of a heat medium, which are two different types of energy. Furthermore, such a conversion is known to have significant losses that cannot be ignored. Likewise, a distribution node indicates that energy flow of an incoming edge in general does not equal the energy flow of an outgoing node. Moreover, the temperatures of the incoming energy flows will be mixed, thus potentially being changed significantly. A heat exchanger node might not alter the magnitude of an energy flow significantly, but primary and secondary temperatures will differ. The magnitude of the energy flow and state variables of the assoiciated fluid flow may be propagated along an energy flow path, but propagation is never allowed across component nodes. The propagated value may be used as estimator for the case of ideal energy transfer, i.e. neglecting heat losses of the pipes. With this propagation mechanism, algorithms can increase the chance of the availability of needed data if propagated values are allowed. Please note that pipes were not included explicitly as component nodes here. For the case of long pipes whose heat losses and temperature drops are never to be ignored, their explicit inclusion as a component node indicates that no forwarding of energy flow and state variables must be made.

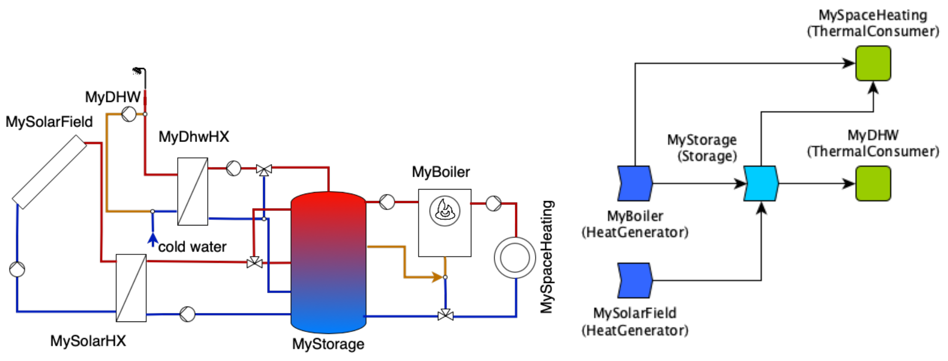

Fig. 2: A (too) simple energy flow graph (right) of a solar thermal combi system (left). Different node colors and shapes indicate different types of nodes. It carries too little information about system structure and allows for ambiguities.

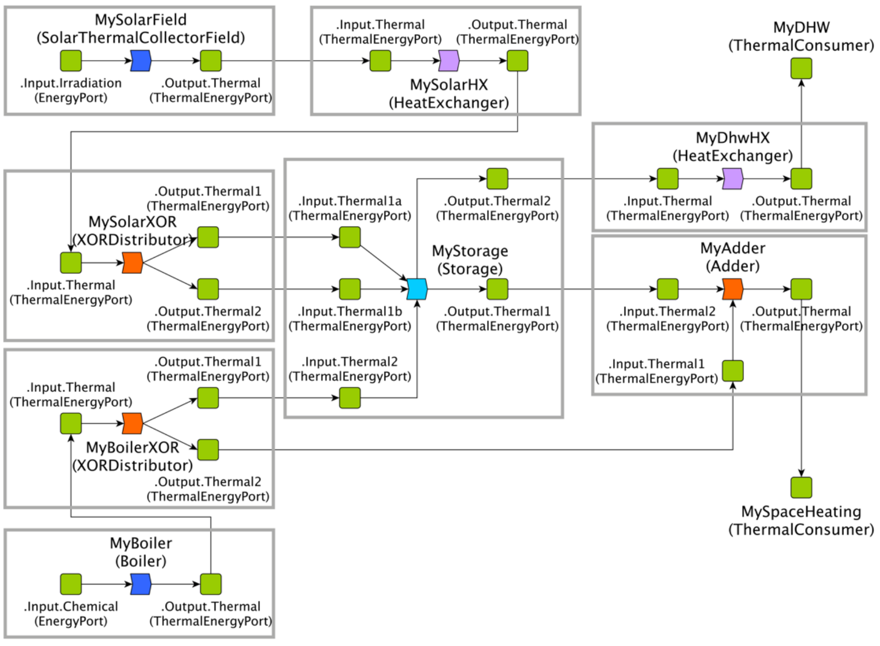
MySolarField (SolarThermalCollectorField)
.Input.Irradiation (EnergyPort)
.Output.Thermal (ThermalEnergyPort)
.Input.Thermal (ThermalEnergyPort)
.Output.Thermal (ThermalEnergyPort)
MySolarHX (HeatExchanger)
MyDHW (ThermalConsumer)
MyDhwHX (HeatExchanger)
.Input.Thermal (ThermalEnergyPort)
.Output.Thermal (ThermalEnergyPort)
MySolarXOR (XORDistributor)
.Output.Thermal1 (ThermalEnergyPort)
.Input.Thermal (ThermalEnergyPort)
.Output.Thermal2 (ThermalEnergyPort)
.Input.Thermal1a (ThermalEnergyPort)
.Output.Thermal2 (ThermalEnergyPort)
MyStorage (Storage)
.Input.Thermal1b (ThermalEnergyPort)
.Output.Thermal1 (ThermalEnergyPort)
.Input.Thermal2 (ThermalEnergyPort)
MyAdder (Adder)
.Input.Thermal2 (ThermalEnergyPort)
.Output.Thermal (ThermalEnergyPort)
.Input.Thermal1 (ThermalEnergyPort)
.Output.Thermal1 (ThermalEnergyPort)
.Input.Thermal (ThermalEnergyPort)
MyBoilerXOR (XORDistributor)
.Output.Thermal2 (ThermalEnergyPort)
MySpaceHeating (ThermalConsumer)
MyBoiler (Boiler)
.Input.Chemical (EnergyPort)
.Output.Thermal (ThermalEnergyPort)

Fig. 3: Extended version of the energy flow graph in fig. 2, with port nodes and additional components. Energy demand for circulation is here included in MyDHW.

The energy flow graph can be created easily by hand with help of predefined subgraphs for the respective components. Only the links between the ports, i.e. the edges between gray boxes in fig. 3., have to be added. The creation and linking process might be automated to a higher degree if existing data about components and piping can be imported from BIM (Building Information Modeling) data in future.

While the energy flow graph is still simple, it already carries much information about the system. Thus, it can be used to describe a system on a basic level.

2.2. Medium flow graph

However, if details of medium flows get important, e.g. in order to trace return flows, a medium flow graph has to be specified. The medium flow graph describes possible flows of different media, e.g. the flow of water in a circuit. To make nodes in a medium flow graph identifiable, further relations and nodes are needed. This is explained for an external heat exchanger subgraph, shown in fig. 4. In order to relate a node to a component, a specific “belongsTo” relation is needed. Contrary to the energy flow graph in fig. 3, the fluid flow cannot be simply directed to a “HX” component node. Otherwise, this would have to be done for both primary and secondary fluid flows. This would effectively connect both fluid circuits to a single connected circuit. Additionally, a relation stating whether a node is on the primary or secondary side is needed. For further refinement, edges in the fluid flow are differentiated into “fluidTransfer” edges (dark blue in fig. 4) and “fluidProcess” edges (light blue in fig. 4). This differentiation adopts the role of the component nodes in the energy flow graph fig. 3. Along “fluidTransfer” edges, state variables may be propagated (as estimators for the case of ideal transfer), while propagation along “fluidProcess” edges is not allowed. In the latter kind of edges, it is indicated that state variables of the fluid change significantly. Thus, nodes at the beginning of a fluidProcess edge represent the state before a significant process (e.g. temperature rise of the fluid), and nodes at the target of fluidProcess edges the state after the process, accordingly.

To unify the graph structures, also the energy flow graph fig. 2 can be modified to contain “energyTransfer” and “energyProcess” edges, such that the permission for value propagation is likewise indicated by the type of the edge and not via the existence of a component node within the energy flow. The component node then is not located within the energy flow, but attached to energyNodes via a “belongsTo” relation.

Similar subgraphes like fig. 3 can be defined for other components like boilers, heat pumps, distributors, etc.

These can be used to quickly create graphs of a system. Thus, only “fluidTransfer” edges between FluidNodes of those subgraphs would have to be added.

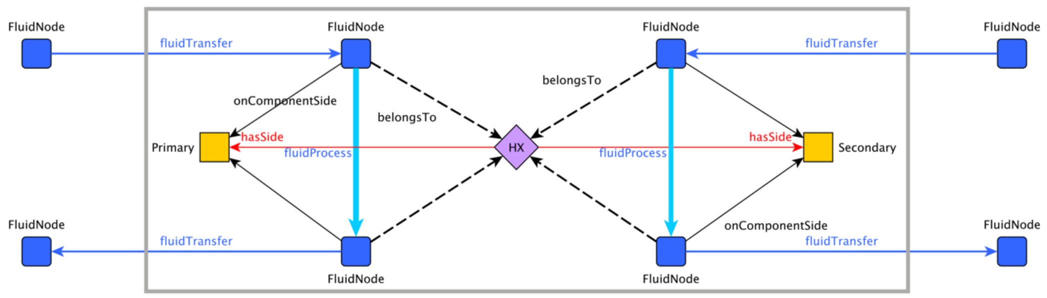

Fig. 4: Subgraph for a heat exchanger. Several nodes and relations are needed to identify nodes in a fluid/medium flow graph.

2.3. Data point identification

The graphs fig. 3 and fig. 4 still miss a connection to available data points, i.e. to actual data of recorded time series of measurement values or logged signals, e.g. the outlet temperature of the heat exchanger at secondary side. Moreover, data points can refer to parameters like storage volume, collector size, etc. In order to assign data points to a fluid, energy or component node, corresponding relations to data point nodes can be defined (compare fig. 5). To simplify data point identification, relative standard data point names can be linked to the data point nodes. For instance, each FluidNode can have data points with associated standard names like “Temperature” or

"VolumetricFlowRate", and each EnergyNode or ComponentNode likewise, allowing to also define standard component parameters. Each standard data point node can then be linked to another "providing" data point node that represents an existing data point that actually can provide data (time series of recorded values or parameter values). It is linked to the system specific unique identifier of that data point. The separation allows to link several standard name data point nodes to a single providing data point, and to explicitly differentiate between a defined and an actually available data point. Furthermore, it is possible to link standard data points to equivalence groups which can store which data points are considered equivalent, independently of the existence of a providing data point. The equivalence groups can be used to later automatically link all contained data points to a providing data point when it is added to the graph.

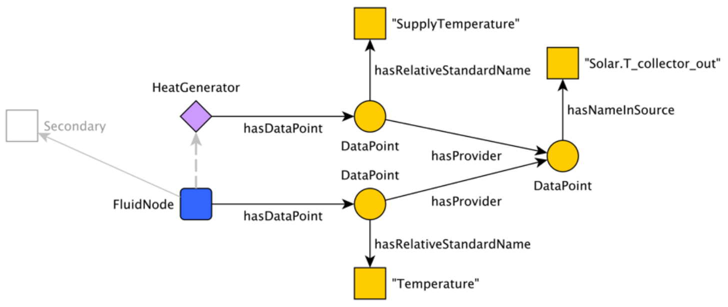

Fig. 5: Data point linking to a node. Actual data can be linked to different nodes with different standard data point names. The gray relations additionally indicate the relation of the "HeatGenerator" and the "FluidNode".

2.4. Component Abstraction levels

From perspective of algorithm authors, it is very useful to request existence of components at different abstraction levels. As described above, it might be sufficient to request a component to be a heat generator, or it might be necessary to require a more specific component like condensing gas boiler because the algorithm uses assumptions or characteristics that are only valid for the more specific component type. For instance, to calculate any coverage ratios, one needs to adress the considered (specific) heat generator as well as "all other heat generators" or "heat consumers" independent of their specific type. By integrating such class hierarchy relations of components into the graph, it is possible to adapt the abstraction level of each (component) node in the request. An example is shown in fig. 6. The graph structure also allows to define several class hierarchies (with other abstraction class nodes) in parallel and thus does not enforce to use a particular taxonomy.

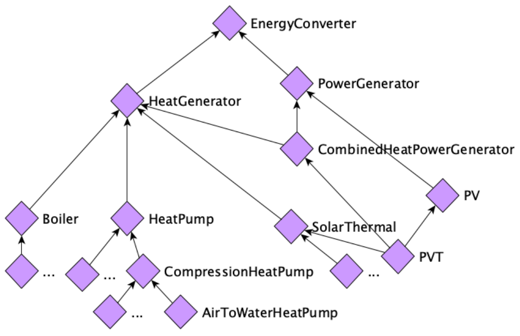

Fig. 6: Exemplary abstraction classes graph. Edges represent the relation "is subclass of" or equivalently "is specialization of".

3. Algorithm selection and linking

If a system is described via a graph with the relations shown in section 2, algorithms can formulate their expected components and relations by defining a requested subgraph structure (see fig. 7). Technically, this can be achieved by storing the system graph as set of RDF (**R**esource **D**escription **F**ramework) triplets, and defining SPARQL (SPARQL **P**rotocol **A**nd **R**DF **Q**uery **L**anguage) queries for each algorithm. SPARQL and RDF are standards defined by the World Wide Web Consortium W3C. It might be necessary to not only check for existence of some relations, but also to check for non-existence of other similar relations (maybe of same type). This is also a reason why in the energy flow graph, explicit “Adder” and “Distributor” nodes are included. In this way, the request can allow or disallow further energy flows (besides the energy flow of interest) to arrive at a component port by simply adding or omitting a corresponding “Adder” node to the request. Without such a node, the number of incoming edges at a port node would have to be checked by different means, complicating the formulation of the algorithm’s requirements.

Furthermore, algorithms might be designed to not execute on each match separately but use a collection of matches as input for a single algorithm instance. For instance, an algorithm might want to cumulate the energies of all heat consumers, needing a collection of all heat consumers as input for a single algorithm instance. Thus, several modes for the process of converting matches to assignments of algorithms (=algorithm instances) must be provided.

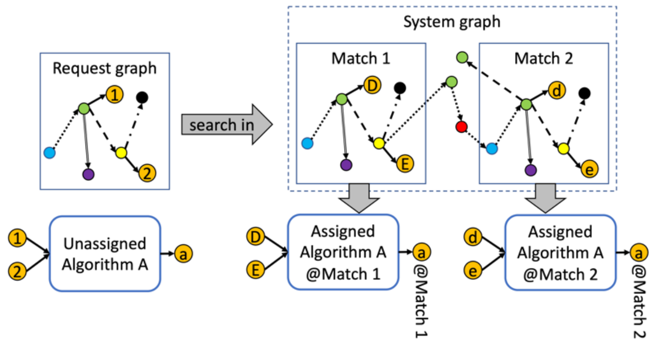

Fig. 7: Illustration for matching an algorithm’s request graph to the system graph. Different node colors and edge styles represent different node types and relations.

4. Conclusion and Outlook

A concept for the description of heating systems as graph for automated FDD has been presented. Advantages of a graph description are:

- Decoupled from specific usage or software solution
- Extensible by different stakeholders, no need to model whole system at once
- Quick compilation from predefined subgraphs
- Easy individual modification and addition of relations
- Standardized file formats for data exchange
- Standardized query language for data retrieval, allowing complex relation patterns to be queried

Further details of the graph description and its usage for fault detection in heating systems will be shown in future work.

5. References

Ohnewein, P., Tschopp, D., Schrammel, H., Gerardts, B., Poier, H., Krammer, S., Köstinger, A., Weinhappl, A., 2016. METHODIQA: Methodik-Entwicklung zur Qualitätssicherung für Erneuerbare Wärme Systeme durch intelligentes Betriebsmonitoring (Endbericht). AEE - Institut für Nachhaltige Technologien (AEE INTEC), Gleisdorf.

VDI Verein Deutscher Ingenieure e.V., 2012. VDI 2169 - Functional checking and yield rating of solar thermal systems. Beuth Verlag GmbH, Berlin.