NANDRAD Modell-Referenz

Andreas Nicolai <andreas.nicolai@tu-dresden.de>, Dirk Weiß, Stephan Hirth,
Katja Tribulowski, Hauke Hirsch, Anne Paepcke

Version 0.2 (22.04.2021)

Inhaltsverzeichnis

1. Überblick

Dieses Dokument enthält eine Beschreibung der verschiedenen Modelle und deren Parametrisierung in der NANDRAD Projektdatei. Dies ist primär eine Eingabereferenz.

Der Abschnitt Struktur der Projektdatei enthält einen Überblick über die Struktur der Projektdatei mit Referenzen zu den einzelnen Abschnitten. Deshalb ist das ein guter Startpunkt, um sich einen Überblick über die NANDRAD Projektdefinition zu verschaffen.

2. NANDRAD Eingabedaten- und Projektdatei-Referenz

2.1. Struktur der Projektdatei

Die NANDRAD-Projektspezifikation ist in einer XML-Datei mit der Erweiterung `nandrad` gespeichert. Der prinzipielle Aufbau der Datei sieht wie folgt aus:

Beispiel 1. Definition einer NANDRAD-Projektdatei

```
<?xml version="1.0" encoding="UTF-8" ?>
<NandradProject fileVersion="2.0">
  <!-- optional DirectoryPlaceholders section-->
  <DirectoryPlaceholders>...</DirectoryPlaceholders>

    <!-- the actual project specification -->
  <Project>
      <ProjectInfo>...</ProjectInfo>
      <Location>...</Location>
      <SimulationParameter>...</SimulationParameter>
      <SolverParameter>...</SolverParameter>
      <Zones>...</Zones>
      <ConstructionInstances>...</ConstructionInstances>
      <HydraulicNetworks>...</HydraulicNetworks>
      <ConstructionTypes>...</ConstructionTypes>
      <Materials>...</Materials>
      <Models>...</Models>
      <Schedules>...</Schedules>
      <Outputs>...</Outputs>
      <ObjectLists>...</ObjectLists>
  </Project>
</NandradProject>
```

Mit dem optionalen `DirectoryPlaceholders` können relative Pfadplatzhalter definiert werden, die für extern referenzierte Dateien verwendet werden sollen (siehe Abschnitt Pfad-Platzhalter).

Alle Projektdaten werden in den `<Project>`-tag eingeschlossen.

Eine Projektdatei kann die folgenden untergeordneten tags enthalten (die Reihenfolge ist beliebig):

XML-Tag	Beschreibung
`ProjectInfo`	Allgemeine Projekt-Meta-Daten → Projektinformationen
`Location`	Klimadaten und Standorteinstellungen → Klimadaten und Standortinformationen

XML-Tag	Beschreibung
SimulationParameter	Allgemeine Parameter des Simulationsmodells → Simulationsparameter
SolverParameter	Einstellungen der numerischen Algorithmen → Solver-Parameter
Zones	Zonenspezifikationen → Zonen
ConstructionInstances	Gebäudekomponenten und Randbedingungen → Konstruktionsinstanzen
HydraulicNetworks	Thermohydraulische Netze → HydraulicNetworks
ConstructionTypes	Datenbank von mehrschichtigen Konstruktionen → Konstruktionstypen
Materials	Materialdatenbank → Materialien
Models	Modell-Parameterblöcke → Modellparametrisierung
Schedules	Definition von geplanten Parametern → Zeitpläne
Outputs	Definition von Ausgaben → Outputs/Ergebnisse
ObjectLists	Definition von Objektlisten/Objektreferenzgruppe → Objektlisten und Ergebnisreferenzen

2.1.1. Eindeutigkeitsforderungen für Definitions-IDs

Im NANDRAD müssen folgenden Objekttypen in einem einzigen ID-Raum definiert werden, d.h. die IDs der unterschiedlichen Objekte dürfen sich nicht doppeln:

- `Zone`
- `ConstructionInstance`
- `EmbeddedObjects`
- `Location.Sensors`

2.2. Grundlegende Datentypen in der NANDRAD-Projektdatei-Spezifikation

Innerhalb der verschiedenen Spezifikationsabschnitte der Projektdatei werden einige grundlegende Datentypen / XML-tags häufig verwendet. Die Regeln für die Spezifikation dieser Parameter sind im Folgenden definiert.

2.2.1. IBK:Parameter

Ein XML-Tag mit dem Namen `IBK:Parameter` definiert einen Fließkommaparameter (floating point value parameter), der durch einen Namen und eine physikalische Einheit identifiziert wird (obligatorische XML-Attribute `name` und `unit`). Der Wert des XML-tags ist der eigentliche Parameterwert.

Beispiel 2. Parameter mit verschiedenen Einheiten

```
<IBK:Parameter name="Volume" unit="m3">30</IBK:Parameter>
<IBK:Parameter name="Temperature" unit="C">20</IBK:Parameter>
<IBK:Parameter name="Temperature" unit="K">293.15</IBK:Parameter>
<!-- unitless parameters take the --- unit -->
<IBK:Parameter name="RelTol" unit="---">0.7</IBK:Parameter>
```

Die Einheiten müssen aus der globalen Einheitenliste ausgewählt werden, siehe Abschnitt Einheitendefinitionen. Wird ein Parameter nicht definiert, wird er als *fehlend* markiert, was bedeutet, dass entweder ein Standardwert verwendet wird oder - im Falle von obligatorischen Benutzerparametern - ein Fehler ausgelöst wird.

2.2.2. IBK:IntPara

Dieser Parameter wird für Flags verwendet. Das obligatorische Attribut `name` identifiziert das Flag. Wird ein Flag nicht definiert, wird es als *fehlend* markiert, was bedeutet, dass entweder ein Standardwert verwendet wird oder - im Falle von obligatorischen Benutzerparametern - ein Fehler ausgelöst wird.

Wird für ganzzahlige Parameter verwendet. Das obligatorische Attribut `Name` identifiziert den Parameter. Der XML-tag `value` ist der Parameterwert. Wird ein Parameter nicht definiert, wird er als *fehlend* markiert, was bedeutet, dass entweder ein Standardwert verwendet wird oder - im Falle von obligatorischen Benutzerparametern - ein Fehler ausgelöst wird.

Beispiel 3. Ganze Zahl (Integer) Parameter-Definition

```
<IBK:IntPara name="DiscMaxElementsPerLayer">30</IBK:IntPara>
```

2.2.3. IBK:Flag

Wird für boolische Schalter (An/Aus-Optionen) verwendet. Das obligatorische Attribut `name` identifiziert das Flag. Wird ein Flag nicht definiert, wird es als *fehlend* markiert, was bedeutet, dass entweder ein Standardwert verwendet wird oder - im Falle von obligatorischen Benutzerparametern - ein Fehler ausgelöst wird.

Beispiel 4. Flag Definition

```
<IBK:Flag name="EnableCyclicSchedules">true</IBK:Flag>
```

Erkannte Werte für Flag-Parameter sind `true` und `1` oder `false` und `0`.

2.2.4. IBK:LinearSpline

Eine linearere Spline ist effektiv eine Datentabelle mit x- und y-Werten, wobei die x-Werte streng monoton steigende Werte sind. Das obligatorische Attribut `name` identifiziert die lineare Spline. Die untergeordneten Tags `X` und `Y` enthalten die tatsächlichen Werte, immer ohne Einheit. Die Anzahl der x- und y-Werte muss übereinstimmen.

Beispiel 5. Lineare Spline-Definition

```
<IBK:LinearSpline name="ThermalLoad">
  <X unit="-">0 6 8 10 17 18 19 20</X>
  <Y unit="-">0 0,5 0,8 1,0 0,7 0,6 0,5 0</Y>
</IBK:LinearSpline>
```

2.2.5. LinearSplineParameter

Ein LinearSpline-Parameter ist effektiv ein erweiterter `IBK:LinearSpline`-Parameter mit zusätzlichen Attributen.

Beispiel 6. LinearSplineParameter Definition

```
<LinearSplineParameter name="ThermalLoad" interpolationMethod="linear">
  <X unit="h">0 6 8 10 17 18 19 20</X>
  <Y unit="W">0 0,5 0,8 1,0 0,7 0,6 0,5 0</Y>
</LinearSplineParameter>
```

Tabelle 1. Attribute

Attribut	Beschreibung	Format	Verwendung
`name`	Spezifischer Name, der sich auf den Raumtyp bezieht, für den der Jahresplan gesetzt wird	string	*required*
`interpolationMethod`	Gibt die Interpolationsmethode zwischen den definierten y-Werten an. • `constant` - konstante Interpolation (Werte konstant während des Zeitschritts) • `linear` - lineare Interpolation (Werte linear interpoliert zwischen Zeitschritten)	Schlüsselwort	*required*
`WrapMethod`	Gibt an, was getan werden soll, wenn Werte mit x-Werten außerhalb des x-Wertebereichs angefordert werden. • `continuous` - konstante Extrapolation (ersten bzw. letzten Wert nehmen) • `cyclic` - zyklische Anpassung mit der modellspezifischen Periodenlänge anwenden (z. B. ein Jahr)	key	*required*

Die Child-Tags `X` und `Y` enthalten jeweils ein obligatorisches Attribut `unit` mit der jeweiligen Werteinheit (siehe Einheitendefinitionen).

Alternativ kann man auch eine Datei mit Tabulator-getrennten Spalten angeben, unter Verwendung des XML-tags `TSVFile`.

Beispiel 7. Linear Spline-Definition mit Angabe der Datei

```
<LinearSplineParameter name="HeatExchangeSpline" interpolationMethod="linear">
   <TSVFile>${Project Directory}/climate/Temperature.csv?3</TSVFile>
</LinearSplineParameter>
```

Beispiel 8. Dazugehörige Datei `Temperature.csv`

```
Time [h]  Temp [C]  otherTemp [C] anotherTemp [C]
0 0 0 0
12  5 7 -9
36  -8  12  65
```

Eine Datei im tsv-Format enthält in der ersten Spalte Zeitwerte und haben danach eine beliebige Anzahl von Datenspalten. Gibt es mehr als eine Datenspalte, muss die Auswahl der Datenspalte durch Anhang des Spezifizierers `?<colIndex>` erfolgen. Die erste Datenspalte hat den Index 1. Daher bezeichnet `?3` wie im Beispiel oben die dritte Spalte (`anotherTemp` im Beispiel oben).

Es ist möglich, Pfad-Platzhalter im Dateinamen zu verwenden (siehe Pfad-Platzhalter).

Man kann entweder `X` UND `Y` angeben, oder alternativ `TSVFile`. Beides ist nicht erlaubt und führt zu einem Fehler.

2.3. Pfad-Platzhalter

In einigen Teilen der NANDRAD-Projektdatei werden externe Dateien referenziert (z.B. Klimadaten-Dateien, siehe Klimadateien). Um den Austausch von Projekten oder Referenzdatendateien in gemeinsamen Datenbankverzeichnissen zu vereinfachen, ist es möglich, Pfadplatzhalter in Dateipfaden zu verwenden.

Sie können z. B. `${MyDatabase}` als `/home/sim/climate_DB` definieren und dann in Ihrem Projekt eine Klimadatendatei referenzieren über `${MyDatabase}/ClimateData.epw`.

Diese Zuordnung der Platzhalter wird zu Beginn der Projektdatei vorgenommen, sodass beim Austausch von Projektdateien zwischen Computern die Platzhalterpfade zu den Verzeichnissen auf dem lokalen Rechner leicht geändert werden können, ohne dass weitere Änderungen in der Projektdatei erforderlich sind.

Die einzelnen Pfadplatzhalter werden in den `DirectoryPlaceholders` definiert:

Beispiel 9. Benutzerdefinierte Directory Placeholders

```
<DirectoryPlaceholders>
  <Placeholder name="Klima DB">/home/sim/climate_DB</Placeholder>
  <Placeholder name="DataFiles">/home/sim/data</Placeholder>
</DirectoryPlaceholder>
```

Es gibt einen eingebauten Platzhalter `${Project Directory}`, der automatisch mit dem Pfad zum Verzeichnis der Projektdatei definiert wird.

2.4. Projektinformationen

Dieser Abschnitt enthält Änderungszeiten/-daten und eine kurze Beschreibung des Projekts. Die folgenden untergeordneten tags werden unterstützt.

Child-Tag	Beschreibung	Format
`Comment`	Allgemeiner Kommentar zum Projekt.	*string*
`Created`	Datum/Uhrzeit der Erstellung dieses Projektes.	*string*
`LastEdited`	Datum/Uhrzeit der letzten Änderung des Projektes.	*string*

Die Datum/Uhrzeit-Strings für `Created` und `LastEdited` sollten das Datum und die Uhrzeit in einem für den Benutzer lesbaren Format speichern, da sie zum Anzeigen von Listen der Projekte mit Änderungs-/Erstellungsdatum verwendet werden können.

2.5. Eingebettete Datenbanken

Um Gebäudekomponenten wie Wände, Decken und Böden usw. zu modellieren, ist es notwendig, einige Parameter für die Materialien zu definieren. Damit ist es dann möglich Konstruktionen zu definieren, die aus solchen Materialien bestehen. Diese Parameter werden in Datenbanken gespeichert, die eigentlich Listen aus XML-Objekten sind.

2.5.1. Materialien

In der NANDRAD-Projektdatei beginnt der Abschnitt der Materialdatenbank mit einem XML-tag namens `materials`.

Beispiel 10. Materialien mit Parametern

```
<Materials>
  <Material id="1001" displayName="Backstein">
    <IBK:Parameter name="Density" unit="kg/m3">2000</IBK:Parameter>
    <IBK:Parameter name="HeatCapacity" unit="J/kgK">1000</IBK:Parameter>
    <IBK:Parameter name="Conductivity" unit="W/mK">1,2</IBK:Parameter>
  </Material>
  <Material id="1004" displayName="Gute Dämmung">
    <IBK:Parameter name="Density" unit="kg/m3">50</IBK:Parameter>
    <IBK:Parameter name="HeatCapacity" unit="J/kgK">1000</IBK:Parameter>
    <IBK:Parameter name="Conductivity" unit="W/mK">0,02</IBK:Parameter>
  </Material>
</Materials>
```

In diesem tag beginnt jedes Materialeigenschaften-Set mit einem XML-tag namens `Material` mit zwei XML-Attributen `id` und `displayName`.

Tabelle 2. Attribute

Attribut	Beschreibung	Format	Verwendung
id	Eindeutige ID des Materials.	> 0	*erforderlich* Informations-/Fehlermeldungen verwendet).
string	*optional*	displayName	Name des Materials (wird für Informations-/Fehlermeldungen verwendet)

Für die Materialparameter wie Dichte, Wärmekapazität und Wärmeleitfähigkeit müssen diese im XML-tag IBK:Parameter definiert werden (siehe IBK:Parameter):

Name	Standardeinheit	Beschreibung	Wertebereich	Verwendung
Density	kg/m3	Trockendichte des Materials.	> 0,01	*erforderlich*
HeatCapacity	J/kgK	Spezifische Wärmekapazität des Materials.	>= 100	*erforderlich*
Conductivity	W/mK	Wärmeleitfähigkeit des trockenen Materials.	>= 1e-5	*erforderlich*

2.5.2. Konstruktionstypen

Konstruktionen werden innerhalb des Abschnitts definiert, der mit einem XML-tag ConstructionTypes beginnt.

Beispiel 11. construction types mit Referenzen zu Materialobjekten

```
<ConstructionTypes>
  <ConstructionTypes id="10005" displayName="Testkonstruktion">
    <MaterialLayers>
      <MaterialLayer thickness="0.2" matId="1001" /> <!-- room side -->
      <MaterialLayer thickness="0.3" matId="1004" />
    </MaterialLayers>
  </ConstructionTypes>
</ConstructionTypes>
```

Innerhalb dieses Abschnitts beginnt jede Konstruktionsdefinition mit dem XML-tag ConstructionType mit den XML-

Attributen `id` und optional `displayName`:

Tabelle 3. Attribute

Attribut	Beschreibung	Format	Verwendung
`id`	Eindeutige Identifikationsnummer	positive Ganzzahl (> 0)	*erforderlich*
`displayName`	Name der Konstruktion (wird für Informations-/Fehlermeldungen verwendet)	string	*optional*

Eine Konstruktion besteht aus einer oder mehreren Materialschichten. Diese werden im untergeordneten XML-tag mit dem Namen `MaterialLayers` definiert. Jede Materialschicht wird mit dem XML-tag `MaterialLayer` mit den folgenden XML-Attributen definiert:

XML-Attribut	Beschreibung	Format	Verwendung
`thickness`	definiert die Dicke der Schicht in `m`	> 0.0	*erforderlich*
`matId`	verweist auf ein Material durch eine eindeutige Material-Identifikationsnummer (`id` wie in einem `Material`-Tag definiert)	string	*erforderlich*

Mit der Verwendung des Attributs *matId* referenzieren Schichten von Konstruktionen die verwendeten Materialien:

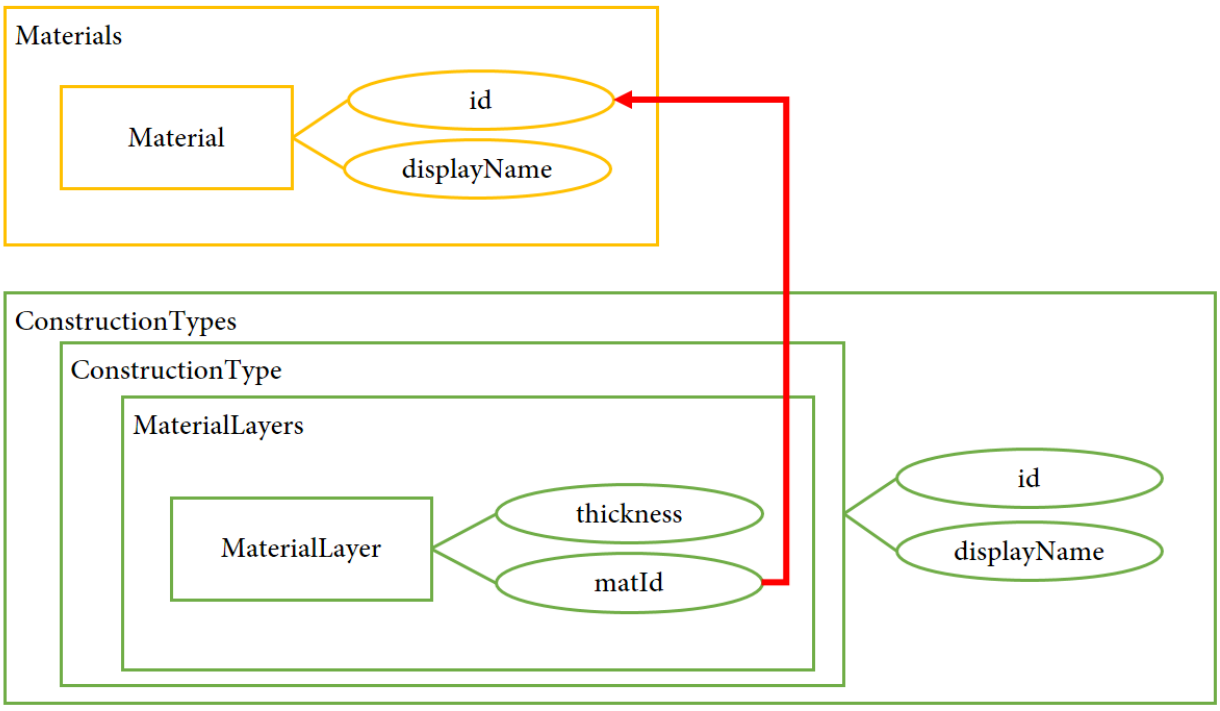


Abbildung 1. Kollaborationsdiagramm für ConstructionType- und Material-Objekte

Das `MaterialLayer` hat keine Child-tags, da alle benötigten Daten wie oben beschrieben als XML-Attribute definiert sind.

Aktive/beheizte Schichten

Jeder Konstruktionsaufbau kann genau eine aktive Schicht haben, welche mit einem Heizregister versehen wird. Falls eine solche Schicht existiert, muss das XML-tag `ActiveLayerIndex` angegeben sein. Dieses XML-Element enthalt den 0-basierten Index der Schicht, d.h. Index 0 entspricht der ersten Schicht in der `MaterialLayers`-Liste.

Falls eine aktive Schicht definiert wurde, muss es irgendwo ein Modell geben, welches passend dafür eine Heizleistung berechnet. Beispielsweise kann dies eine Fußbodenheizung sein (siehe [network_floor_heating]).

2.5.3. Verglasungssysteme

Verglasungssysteme werden in einer Liste innerhalb des XML-tags `WindowGlazingSystems` definiert.

Beispiel 12. Parameterdefinition für ein Verglasungssystem

```
<WindowGlazingSystems>
  <WindowGlazingSystem id="123" modelType="Standard">
    <IBK:Parameter name="ThermalTransmittance" unit="W/m2K">0,4</IBK:Parameter>
    <LinearSplineParameter name="SHGC" interpolationMethod="linear" wrapMethod="cyclic">
      <!-- X incidence angle - 90 deg = Sonne steht senkrecht/normal zur Oberfläche -->
      <X unit="Deg">0 90 </X>
      <!-- Hinweis: kein konstanter Parameter - SHGC konstant wird wie unten definiert -->
      <Y unit="---">0.6 0.6 </Y>
    </LinearSplineParameter>
  </WindowGlazingSystem>
</WindowGlazingSystems>
```

Innerhalb dieses Abschnitts beginnt jede Definition eines Verglasungssystem mit dem XML-tag `WindowGlazingSystem` mit den XML-Attributen `id`, `modelType` und optional `displayName`:

Tabelle 4. Attribute

Attribut	Beschreibung	Format	Verwendung
`id`	Eindeutige Identifikationsnummer	positive integer (> 0)	*erforderlich*
`displayName`	Name des Verglasungssystems (wird für Informations-/Fehlermeldungen verwendet)	string	*optional*
`modelType`	Identifiziert die Modellkomplexität: • `Standard` - Standard-Verglasungsmodell, mit einem U-Wert (Wärmedurchgangskoeffizient) und einfallswinkelabhängigem SHGC-Wert	string	*optional*

Skalare Parameter werden innerhalb eines XML-tags `IBK:Parameter` definiert (siehe IBK:Parameter):

Name	Standardeinheit	Beschreibung	Wertebereich	Verwendung
ThermalTransmittance	W/m2K	Wärmedurchgangskoeffizient der Verglasung	> 0	*erforderlich für Modelltyp Simple*

Parameter, die vom Einfallswinkel abhängen, werden in einem XML-tag `LinearSplineParameter` definiert (siehe LinearSplineParameter):

Name	Standardeinheit	Beschreibung	Wertebereich	Verwendung
SHGC	---	Solarer Wärmegewinnkoeffizient	> 0	*erforderlich für Modelltyp Simple*

2.6. Zonen

Um Gebäude modellieren zu können, ist es notwendig die einzelnen Räume mit den entsprechenden Parametern zu definieren. Eine Zone definiert einen thermisch gut durchmischten Bereich/Raum mit einer einzigen/einheitlichen Lufttemperatur.

Objekte vom Typ `Zone` speichern alle Eigenschaften die benötigt werden, um die Zonentemperatur aus der Energiedichte (der Erhaltungsgröße) zu berechnen.

Beispiel 13. Definition der Zonen

```
<Zones>
  <Zone id="1" displayName="Var01" type="Active">
    <IBK:Parameter name="Area" unit="m2">10</IBK:Parameter>
    <IBK:Parameter name="Volume" unit="m3">30</IBK:Parameter>
  </Zone>
</Zones>
```

Innerhalb des XML-tags namens `Zones` beginnt jede Zone mit dem XML-tag `Zone`. Die folgenden XML-Attribute müssen definiert werden:

```
<Zone id="1" displayName="Var01" type="Active">
```

Tabelle 5. Attribute

Attribut	Beschreibung	Format	Verwendung
id	eindeutige Identifikationsnumer der Zone	(> 0)	*erforderlich*

Attribut	Beschreibung	Format	Verwendung
displayName	Anzeigename der Zone. Wird benötigt, um die Zone im Datenmodell und in Ausgaben leichter zu finden.	string	*optional*
type	Legt fest, ob die Zone ausgeglichen und in das Gleichungssystem einbezogen wird. • Constant als Zone mit konstanten/vordefinierten Temperaturen (Zeitplan) • Active als Zone, die durch einen Temperaturknoten im Raum beschrieben wird • Ground als Bodenzone (berechnet die Temperatur auf Basis des Standards)	key	*erforderlich*

Für *konstante* Zonen wird angenommen, dass die Temperatur vorgegeben ist, während in *Active* Zonen die Temperatur berechnet wird (d. h. in den Unbekannten des Modells enthalten ist). Eine *konstante* Zone benötigt nur den Temperaturparameter.

Parameter (siehe Abschnitt IBK:Parameter für eine Beschreibung des tags IBK:Parameter):

Name	Einheit	Beschreibung	Wertebereich	Verwendung
Volume	m3	Zonenluftvolumen	> 0.0	*erforderlich*
Area	m2	Nettonutzfläche des Erdgeschosses (für flächenbezogene Leistungen und Lasten)	> 0.0	*optional*
HeatCapacity	J/K	Zusätzliche Wärmekapazität (Möbel, etc.)	>= 0.0	*optional*
Temperature	C	Temperatur der Zone nur verwendet, wenn constant	-70...120	(*erforderlich*)
RelativeHumidity	%	Relative Luftfeuchtigkeit der Zone nur verwendet, wenn constant	0...100	(*erforderlich*)
CO2Concentration	g/m3	CO2-Konzentration der Zone nur verwendet, wenn constant	> 0.0	(*erforderlich*)

Die Parameter RelativeHumidity und CO2Concentration müssen nur für *konstante* Zonen definiert werden, wenn die jeweilige Bilanzgleichung aktiviert ist.

2.7. Konstruktionsinstanzen

Konstruktionsinstanzen repräsentieren tatsächlich verbaute eindimensionale Teile der Gebäudehülle, z.B. Wände, Böden, Decken, Dächer.

Beispiel 14. Definition einer Außenwand nur mit der Randbedingung Wärmeleitung

```
<ConstructionInstances>
  <!-- Oberfläche Var 01 -->
  <ConstructionInstances id="1" displayName="All Surfaces Var01">
    <ConstructionTypeId>10005</ConstructionTypeId>
    <IBK:Parameter name="Area" unit="m2">62</IBK:Parameter>
    <InterfaceA id="10" zoneId="1">
      <!--Interface zu 'Room'-->
      <InterfaceHeatConduction modelType="Constant">
        <IBK:Parameter name="HeatTransferCoefficient" unit="W/m2K">2.5</IBK:Parameter>
      </InterfaceHeatConduction>
    </InterfaceA>
    <InterfaceB id="11" zoneId="0">
      <!-Schnittstelle nach außen-->
      <InterfaceHeatConduction modelType="Constant">
        <IBK:Parameter name="HeatTransferCoefficient" unit="W/m2K">8</IBK:Parameter>
      </InterfaceHeatConduction>
    </InterfaceB>
  </ConstructionInstances>
</ConstructionInstances>
```

Die Konstruktionsinstanzen werden innerhalb des XML-tags `ConstructionInstances` definiert. Innerhalb des Abschnitts beginnt jede Konstruktionsdefinition mit dem XML-tag `ConstructionInstance` mit den Attributen `id` und `displayName`.

Tabelle 6. Attribute

Attribut	Beschreibung	Format	Verwendung
`id`	Identifikationsnummer der Konstruktionsinstanz	> 0	*erforderlich*
`displayName`	Anzeigename der Konstruktionsinstanz. Wird benötigt, um die Konstruktionsinstanz im Datenmodell und in der Ausgaben leichter zu finden.	string	*optional*

Die Konstruktionsinstanz hat das folgende *erforderliche* Child-tags:

Tabelle 7. Construction Instance Child tags

Tag	Beschreibung
`ConstructionTypeId`	Referenz auf `ConstructionTypeId`
`IBK:Parameter`	Verschiedene `IBK:Parameter` für Konstruktionsinstanz
`InterfaceA`	Schnittstelle für Konstruktionsinstanz Seite A
`InterfaceB`	Interface für Konstruktionen

`ConstructionTypeId` ist eindeutige Id, die den Konstruktionstyp (Schichtenaufbau) der Konstruktionsinstanz definiert.

Für die Parameter der Konstruktionsinstanz können die folgenden XML-tags mit dem Namen `IBK:Parameters` mit den XML-Attributen `name` und `unit` mit den folgenden Einträgen definiert werden:

Name	Einheit	Beschreibung	Wertebereich	Verwendung
`Orientation`	Deg	Ausrichtung der Wand wenn eine Schnittstelle eine solare (kurzwellige) Strahlungs-Randbedingung hat, ist sie *erforderlich*	0...360	*erforderlich* / *optional*
`Inclination`	Deg	Neigung der Wand • 0 Deg - Dach • 90 Grad - senkrechte Wand • 180 Deg - nach unten gerichtet wenn eine Schnittstelle kurz- und/oder langwellige Strahlungsrandbedingung hat, ist sie *erforderlich*	0...180	*erforderlich* / *optional*
`Area`	m2	Bruttofläche der Wand (inkl. evtl. vorhandener Fenster, Löcher etc.)	> 0	*erforderlich*

Darin müssen die Schnittstellen mit dem XML-tag `InterfaceA` und `InterfaceB` angegeben werden. Schließlich müssen die Interfaces mit dem XML-tag `InterfaceA` und `InterfaceB` mit den XML-Attributen `id` und `zoneId` definiert werden. Im Folgenden wird dies im Detail beschrieben.

2.7.1. Räumliche Diskretisierung (Finite-Volumen-Methode)

Während der Berechnung wird jede der Konstruktionen mit Hilfe eines Algorithmus zur Gittergenerierung räumlich diskretisiert. Dieser Algorithmus verwendet drei einflussreiche Parameter, die im Abschnitt Solver-Parameter definiert sind:

- `DiscMinDx`
- `DiscStretchFactor`
- `DiscMaxElementsPerLayer`

Abbildung 2 veranschaulicht die Wirkung verschiedener Dehnungsfaktoren

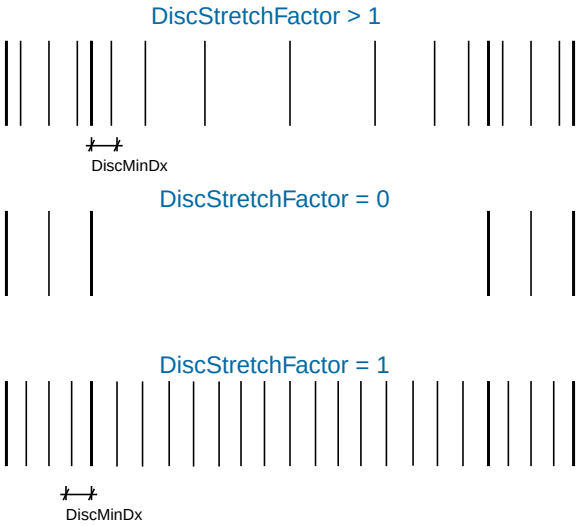


Abbildung 2. Verschiedene Diskretisierungsvarianten in Abhängigkeit vom Parameter DiscStretchFactor

Grundsätzlich werden drei verschiedene Gittergenerierungsverfahren unterstützt:

- **minimal grid:** bei `DiscStretchFactor = 0` erzeugt der Algorithmus ein Finites Volumen pro Materialschicht, mit Ausnahme der Randelemente, die immer in zwei aufgeteilt werden (notwendig für die Oberflächenwertextrapolation). So ergeben sich z. B. bei einem 4-Schicht-Aufbau 6 Finite Volumen.
- **equidistant:** bei `DiscStretchFactor = 1` erzeugt der Algorithmus in jeder Schicht gleichmäßig verteilte Gitterelemente, deren Dicke nahe, aber immer kleiner als der Parameter `DiscMinDx` ist. Da Materialschichten unterschiedliche Breiten haben können, ist eine einheitliche Dicke der Gitterelemente in der gesamten Konstruktion möglicherweise nicht möglich. Wählen Sie einen `DiscMinDx`-Parameter, bei dem alle Materialschichtbreiten ganzzahlige Vielfache dieser Rasterelementdicke sind (z.B. *1 mm*)
- **regular grid:** für jeden `DiscStretchFactor > 1` wird ein regelmäßiges, variabel beabstandetes Gitter erzeugt.

Algorithmus zur Erzeugung eines regulären Gitters

Ein regelmäßiges Streckgitter wird mit einer doppelseitigen *tanh*-Streckfunktion erzeugt. Der Faktor `DiscStretchFactor` bestimmt dabei ungefähr das Verhältnis der ersten beiden Gitterelementbreiten. Natürlich variiert dieser Wachstumsfaktor und geht in der Mitte einer Materialschicht gegen Null, aber er bestimmt sehr schön den gesamten Gitterausschnitt. Ein Faktor von 4 ist ein guter Standardwert.

Der Parameter `DiscMinDx` definiert die maximale Breite der äußersten Gitterelemente in jeder Schicht. Damit wird indirekt auch die Anzahl der Gitterelemente pro Materialschicht bestimmt. Mit zunehmender Anzahl von Gitterelementen pro Schicht werden die äußersten Gitterelemente kleiner. Auf diese Weise bestimmt der Algorithmus die Anzahl der Gitterzellen (für einen gegebenen `DiscStretchFactor`), bis die erzeugte Breite bei den äußersten Gitterelementen gleich oder kleiner als der Parameter `DiscMinDx` ist. Eine minimale Elementdicke von *2 mm* ist ein guter Standardwert für sehr genaue Berechnungen, aber ein Wert von *5 mm* kann in vielen Situationen ausreichen (dies reduziert die Anzahl der Unbekannten und eventuell die Simulationszeit erheblich).

Schließlich gibt es noch den Parameter `DiscMaxElementsPerLayer`, mit dem die Anzahl der zu erzeugenden Gitterelemente in einer Materialschicht begrenzt werden kann. Dies ist besonders dann sinnvoll, wenn sehr dicke Materialschichten vorhanden sind und eine große Anzahl von Gitterzellen erzeugt wird. Oft wird diese Genauigkeit

nicht benötigt (jedenfalls bei sehr dicken Materialschichten), so dass eine Begrenzung der Anzahl zur Beschleunigung der Berechnung sinnvoll sein kann. Solange die Anzahl der erzeugten Gitterzellen pro Materialschicht `DiscMaxElementsPerLayer` überschreitet, wird der Algorithmus den `DiscStretchFactor` schrittweise erhöhen, bis das Kriterium erfüllt ist. Der Solver wird für jede Konstruktionsschicht, auf die diese Anpassung angewendet wird, eine Warnmeldung ausgeben.

Wie bei allen numerischen Lösern, die mit Rechengittern arbeiten, gibt es immer einen Kompromiss zwischen Geschwindigkeit und Genauigkeit. Eine Studie über die Empfindlichkeit des Gitters kann hilfreich sein, z. B. indem Sie mit `DiscMinDx = 5 mm` und `DiscStretchFactor = 8` beginnen und dann die Werte schrittweise reduzieren, bis sich die Lösung nicht mehr verändert. Für kleine Gebäude/Modelle, bei denen die Leistung keine Rolle spielt, können die Standardwerte `DiscMinDx = 2 mm` und `DiscStretchFactor = 4` verwendet werden.

2.8. Interfaces (Konstruktions-Randbedingungen)

Die Interfaces definieren Randbedingungen und Parameter für die ein oder zwei Oberflächen `InterfaceA` und `InterfaceB` einer Konstruktionsinstanz. Wenn die Konstruktionsinstanz eine adiabatische Wand definiert, wird nur ein Interface benötigt. In allen anderen Fällen werden zwei Schnittstellen benötigt. Das `InterfaceA` verknüpft die erste Materialschicht aus dem Konstruktionstyp mit der zugeordneten Zone über die `zoneId`. Das `InterfaceB` verknüpft die letzte Materialschicht aus dem Konstruktionstyp mit der `zoneId` von `InterfaceB`.

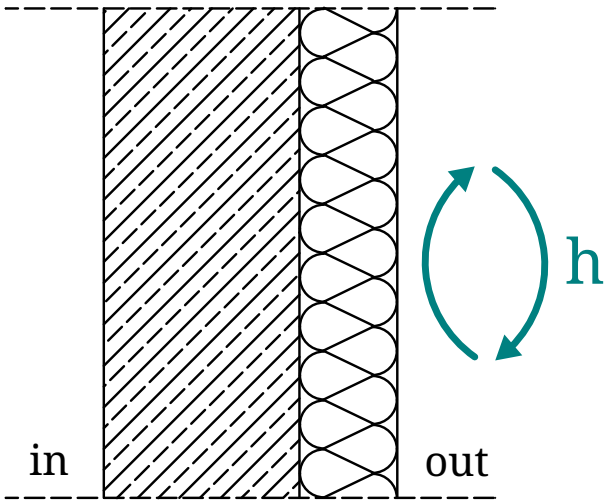
Beispiel 15. Schnittstellendefinitionen für eine Konstruktion mit Schnittstellen für beide Seiten

```
<ConstructionInstance id="1" displayName="All Surfaces Var01">
    ...
    <InterfaceA id="10" zoneId="1">
      <InterfaceHeatConduction modelType="Constant">
        <IBK:Parameter name="HeatTransferCoefficient" unit="W/m2K">2.5</IBK:Parameter>
      </InterfaceHeatConduction>
    </InterfaceA>
    <InterfaceB id="11" zoneId="0">
      <InterfaceHeatConduction modelType="Constant">
        <IBK:Parameter name="HeatTransferCoefficient" unit="W/m2K">8</IBK:Parameter>
      </InterfaceHeatConduction>
        <InterfaceSolarAbsorption model="Constant">
            <IBK:Parameter name="AbsorptionCoefficient" unit="---">0,6</IBK:Parameter>
        </InterfaceSolarAbsorption>
        <InterfaceLongWaveEmission model="Constant">
            <IBK:Parameter name="Emissivity" unit="---">0,9</IBK:Parameter>
        </InterfaceLongWaveEmission>
    </InterfaceB>
</ConstructionInstance>
```

`InterfaceA` und `InterfaceB` können ein oder mehrere untergeordnete tags haben.

2.8.1. Wärmeleitung

Die konvektive Wärmeleitung über die Schnittstelle wird durch das XML-tag `InterfaceHeatConduction` beschrieben.



Beispiel 16. Parameterdefinition für die Randbedingung Wärmeleitung

```
<InterfaceHeatConduction modelType="Constant">
  <IBK:Parameter name="HeatTransferCoefficient" unit="W/m2K">2.5</IBK:Parameter>
</InterfaceHeatConduction>
```

Die `InterfaceHeatConduction` muss mit dem folgenden XML-Attribut `modelType` definiert werden.

Tabelle 8. Attribute

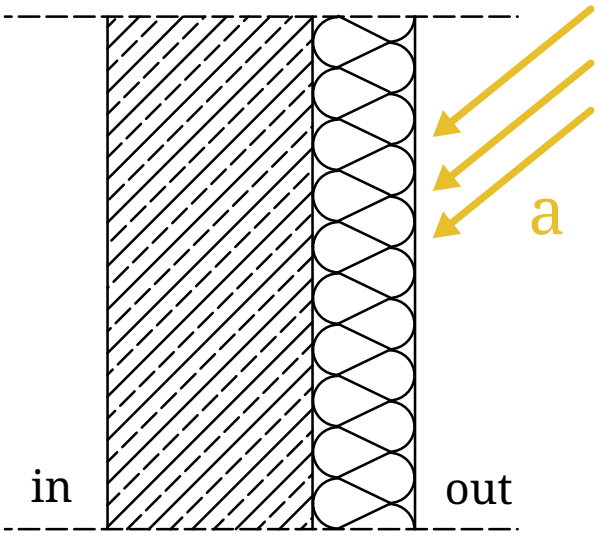
Attribut	Beschreibung	Format	Verwendung
`modelType`	Setzt den Typ des Wärmeleitungsmodells • `Constant` - es wird ein konstantes Modell verwendet (derzeit die einzige Option)	key	*erforderlich*

Fließkommaparameter (siehe Abschnitt IBK:Parameter für eine Beschreibung des tags `IBK:Parameter`):

Name	Vorgabeeinheit	Beschreibung	Wertebereich	Verwendung
`HeatTransferCoefficient`	W/m2K	Konstanter konvektiver Wärmeübergangskoeffizient	> 0.0	*erforderlich*

2.8.2. Solare Absorption

Die solare Absorption über die Schnittstelle wird durch das XML-tag `InterfaceSolarAbsorption` beschrieben. Dieser Koeffizient beschreibt die solare Kurzwellenstrahlung, die von der Grenzfläche absorbiert wird.



Beispiel 17. Parameterdefinition für die Randbedingung Solare Absorption

```
<InterfaceSolarAbsorption modelType="Constant">
  <IBK:Parameter name="AbsorptionCoefficient" unit="---">0.6</IBK:Parameter>
</InterfaceHeatConduction>
```

Das `InterfaceSolarAbsorption` muss mit dem folgenden XML-Attribut `modelType` definiert werden.

Tabelle 9. Attribute

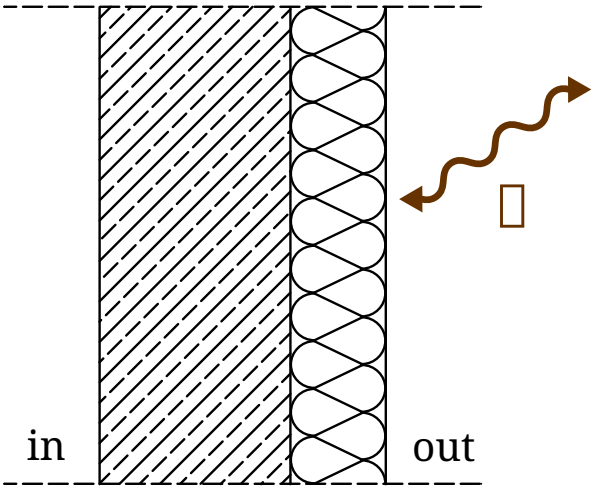
Attribut	Beschreibung	Format	Verwendung
`modellType`	Setzt den Typ des Wärmeleitungsmodells • `Constant` - es wird ein konstantes Modell verwendet (derzeit die einzige Option)	key	*erforderlich*

Es können XML-tags mit dem Namen `IBK:Parameter` mit den XML-Attributen `name` und `unit` mit den folgenden Einträgen definiert werden:

Name	Einheit	Beschreibung	Wertebereich	Verwendung
`AbsorptionCoefficient`	---	Konstanter Absorptionskoeffizient	0...1	*erforderlich*

2.8.3. Langwellige Emission

Die langwellige Emission über die Schnittstelle wird durch das XML-tag `InterfaceLongWaveEmission` beschrieben. Dieser Koeffizient beschreibt die langwellige Absorption und Emission über die Schnittstelle.



Beispiel 18. parameterdefinition für langwellige Emission

```
<InterfaceLongWaveEmission modelType="Constant">
  <IBK:Parameter name="Emissivity" unit="---">0,9</IBK:Parameter>
</InterfaceLongWaveEmission>
```

Die `InterfaceLongWaveEmission` muss mit dem folgenden XML-Attribut `modelType` definiert werden.

Tabelle 10. Attribute

Attribut	Beschreibung	Format	Verwendung
`modellType`	Setzt den Typ des Wärmeleitungsmodells • `Constant` - es wird ein konstantes Modell verwendet (derzeit die einzige Option)	key	*erforderlich*

Es können XML-tags mit dem Namen `IBK:Parameter` mit den XML-Attributen `name` und `unit` mit den folgenden Einträgen definiert werden:

Name	Einheit	Beschreibung	Wertebereich	Verwendung
`Emissivity`	---	Konstanter Absorptionskoeffizient	0...1	*erforderlich*

2.8.4. Dampfdiffusion

MUSS SPÄTER DEFINIERT WERDEN.

Die Dampfdiffusion über die Grenzfläche wird durch das XML-tag `InterfaceVaporDiffusion` beschrieben.

Beispiel 19. Parameterdefinition für Dampfdiffusion

```
<InterfaceVaporDiffusion modelType="Constant">
  <IBK:Parameter name="VaporTransferCoefficient" unit="s/m">1</IBK:Parameter>
</InterfaceVaporDiffusion>
```

Das `InterfaceVaporDiffusion` muss mit dem folgenden XML-Attribut `modelType` definiert werden.

Tabelle 11. Parameter für das InterfaceVaporDiffusion-tag

Attribut	Beschreibung	Format	Verwendung
`modellType`	Setzt den Typ des Wärmeleitungsmodells • `Constant` - es wird ein konstantes Modell verwendet (derzeit die einzige Option)	key	*erforderlich*

Es können XML-tags mit dem Namen `IBK:Parameter` mit den XML-Attributen `name` und `unit` mit den folgenden Einträgen definiert werden:

Name	Einheit	Beschreibung	Wertebereich	Verwendung
`VaporTransferCoefficient`	s/m	Dampfübergangskoeffizient	> 0.0	*erforderlich*

2.8.5. Luftstrom

MUSS SPÄTER DEFINIERT WERDEN.

Der Luftstrom über die Schnittstelle wird mit einem Druckkoeffizienten berechnet. Er wird im XML-tag `InterfaceAirFlow` beschrieben.

Beispiel 20. Parameterdefinition für Luftstrom

```
<InterfaceAirFlow modelType="Constant">
  <IBK:Parameter name="PressureCoefficient" unit="---">0.6</IBK:Parameter>
</InterfaceAirFlow>
```

Das `InterfaceAirFlow` muss mit dem folgenden XML-Attribut `modelType` definiert werden.

Tabelle 12. Attriubute

Attribut	Beschreibung	Format	Verwendung
modelType	Setzt den Typ des Luftstroms • Constant - es wird ein konstantes Modell verwendet (derzeit die einzige Option)	key	*erforderlich*

Es können XML-tags mit dem Namen IBK:Parameter mit den XML-Attributen name und unit mit den folgenden Einträgen definiert werden:

Name	Einheit	Beschreibung	Werteberich	Verwendung
PressureCoefficient	---	Druckkoeffizient	0...1	*erforderlich*

2.9. Aktive Schichten/Flächenheizungen

Eine Konstruktion kann thermisch wechselwirken mit andern Modelle, bspw. als Fußbodenheizung. Dafür muss im verwendeten Konstruktionstyp eine aktive Schicht definiert sein (siehe Section 2.5.2.1).

2.10. Eingebettete Objekte (Fenster, Türen, Öffnungen...)

Es kann mehrere Definitionen für eingebettete Objekte geben.

Beispiel 21. Definition eines Fensters innerhalb einer Bauinstanz

```
<ConstructionInstance id="1">
  <IBK:Parameter name="Area" unit="m2">12</IBK:Parameter>
    ...
  <EmbeddedObjects>
    <EmbeddedObject id="2000" displayName="Ein Fenster">
        <!-- Area-Parameter ist erforderlich. -->
      <IBK:Parameter name="Area" unit="m2">8</IBK:Parameter>
        ...
    </EmbeddedObject>
  </EmbeddedObjects>
</ConstructionInstance>
```

Eingebettete Objekte müssen mindestens einen Parameter Area definiert haben. Diese Fläche darf die Bruttofläche der Konstruktionsinstanz nicht überschreiten.

Ein eingebettetes Objekt wird durch eingebettete Datenobjekte weiter qualifiziert.

2.10.1. Fenster

Ein Fenster besteht aus einer Verglasung und optional einem Rahmen und Trennwänden. Ohne Rahmen und Trennwände sieht die Definition für ein solches Fenster wie folgt aus:

Beispiel 22. Parameterdefinition für Basisfenster ohne Rahmen

```
<EmbeddedObject id="2000" displayName="Ein Fenster">
  <IBK:Parameter name="Area" unit="m2">8</IBK:Parameter>
    <Window glazingSystemID="123"/>
</EmbeddedObject>
```

Nur das Verglasungssystem wird über die ID referenziert. Verglasungssysteme sind in der Datenbankliste der Verglasungssysteme definiert, siehe Section 2.5.3.

Das Fenster kann einen Rahmen und/oder Trennwände haben. Diese sind separate Entitäten, da das Material von Rahmen und Trennwänden (und damit die Wärmeleitfähigkeit zwischen diesen Materialien) unterschiedlich sein kann. Diese werden in den XML-tags `Frame` und `Divider` definiert:

Beispiel 23. Parameterdefinition für Basisfenster mit Rahmen und Trennwand

```
<EmbeddedObject id="2000" displayName="Ein Fenster">
  <IBK:Parameter name="Area" unit="m2">8</IBK:Parameter>
  <Window glazingSystemID="123">
    <Frame materialID="1001">
      <IBK:Parameter name="Area" unit="m2">3</IBK:Parameter>
    </Frame>
    <Divider materialID="1002">
      <IBK:Parameter name="Area" unit="m2">2</IBK:Parameter>
    </Divider>
  </Window>
</EmbeddedObject>
```

Die Materialeigenschaften (derzeit nur die Wärmeleitfähigkeit) von Rahmen- und Trennelementen werden aus dem über die ID referenzierten Material übernommen.

Die tatsächliche Geometrie von Rahmen- und Trennelementen ist nicht wichtig, aber ihre Gesamtquerschnittsfläche muss als Parameter `Area` angegeben werden.

Der von Rahmen und Trennwand belegte Querschnitt darf die Bruttofläche des eingebetteten Fensterobjekts nicht überschreiten. Die tatsächliche lichtdurchlässige Verglasungsfläche wird als Differenz zwischen der Fläche des eingebetteten Objekts und den Flächen von Rahmen und Trennwand berechnet.

Wenn die Größe des Fensters (oder des eingebetteten Objekts) geändert wird, müssen die Größen von Rahmen und Trennwand entsprechend angepasst werden. Es wäre zwar möglich gewesen, Rahmen- und Trennwandquerschnitte auch als relativen Prozentsatz zu definieren, dennoch muss dieser Prozentsatz bei einer Größenänderung des Fensters aktualisiert werden.

Fensterverschattung

Es ist möglich, vorberechnete Verschattung bzw. Verschattungseinrichtungen sowohl auf opake als auch auf transluzente Fassadenelemente anzuwenden. Dabei wird zwischen geregelter/fester Verschattungsvorrichtung am Fenster und einer geometrischen Umgebungs-/Eigenverschattung unterschieden.

Die vorberechnete Umgebungsverschattung bzw. Eigenverschattung wird als globale Eigenschaft im `Location`-tag definiert (siehe Section 2.11.4).

Wenn eine vorberechnete Umgebungsverschattung definiert ist, wird für **jede** opake und transluzente Fläche ein Verschattungsgrad (Abminderungsfaktor) angegeben. Dieser wird automatisch bei der Strahlungsberechnung auf Flächen und Fenster einbezogen.

Die nachfolgend beschriebene (geregelte) Fensterverschattung wird **zusätzlich** berücksichtigt.

Wie im Abschnitt Section 2.11.4 beschrieben, erfolgt die Zuordnung zwischen bereitgestellten Datenspalten und Objekt-ID über die eindeutige ID Nummer der Konstruktionsinstanz bzw. des eingebetteten Objekts.

Alternativ oder zusätzlich zur vorberechneten Umgebungsverschattung ist es möglich, eine geregelte Verschattung für das Fenster zu definieren.

Beispiel 24. Parameterdefinition für konstante Verschattung

```
<Window glazingSystemID="123">
    ...
  <Shading modelType="Constant">
    <IBK:Parameter name="ReductionFactor" unit="---">0.6</IBK:Parameter>
  </Shading>
</Window>
```

Das XML-tag `Shading` muss mit den folgenden XML-Attributen definiert werden:

Tabelle 13. Attribute

Attribut	Beschreibung	Format	Verwendung
`modelType`	Setzt den Typ des Schattierungsmodells • `Constant` - Konstante Verschattung • `Precomputed` - Zeitabhängige vorberechnete Verschattungsfaktoren • `Controlled` - Verschattung wird in Abhängigkeit einer Strahlungsintensität	key	*erforderlich*
`controlModelID`	ID des Verschattungskontrollmodells	ID	*erforderlich für* `Controlled`

Tabelle 14. Child-Tags

Element	Beschreibung	Format	Verwendung
PrecomputedReductionFactor	Zeitreihe mit vorberechneten Abminderungsfaktoren infolge Verschattung (sollte für eine vorberechnete, geregelte Verschattung verwendet werden)		*erforderlich für* Precomputed

Es können XML-tags mit dem Namen IBK:Parameters mit den XML-Attributen name und unit mit den folgenden Einträgen definiert werden:

Name	Einheit	Beschreibung	Wertebereich	Verwendung
ReductionFactor	---	Prozentualer Anteil der verbleibenden solaren Gewinne, wenn die Beschattung geschlossen ist	0...1	*erforderlich für* Constant und Controlled

Berechnung des Beschattungsfaktors auf Basis des Steuersignals

```
Nominale (maximale) ReductionFactor = z = 80%

z-Wert in Abhängigkeit vom Steuersignal Fz:

Fz = 1   = voll beschattet:           z = 1 - (1 - 80%) * Fz = 0,8
Fz = 0   = unverschattet verschattet: z = 1 - (1 - 80%) * Fz = 1
Fz = 0,5 = teilweise verschattet:     z = 1 - (1 - 80%) * Fz = 0,9
```

2.11. Klimadaten und Standortinformationen

2.11.1. Übersicht

Klimalasten werden in NANDRAD über Klimadateien bereitgestellt. Für die Berechnung der Sonneneinstrahlung werden Informationen über den Gebäudestandort (in der Regel in der Klimadatei enthalten) sowie die Ausrichtung und Neigung der verschiedenen Konstruktionsflächen benötigt (definiert für Außenflächen, siehe Section 2.7).

2.11.2. Spezifikation

Informationen über Standort- und Klimadaten werden im Abschnitt Location der Projektdatei gespeichert:

Beispiel 25. Definition des Standorts

```
<Location>
  <ClimateFilePath>${Projektverzeichnis}/climate/GER_Potsdam_2017.c6b</ClimateFilePath>
  <IBK:Parameter name="Latitude" unit="Deg">51</IBK:Parameter>
  <IBK:Parameter name="Longitude" unit="Deg">13</IBK:Parameter>
  <IBK:Parameter name="Albedo" unit="---">0.2</IBK:Parameter>
  <IBK:Parameter name="Altitude" unit="m">100</IBK:Parameter>
  <IBK:Flag name="PerezDiffuseRadiationModel">false</IBK:Flag>
</Location>
```

Die Außenklimabedingungen, einschließlich Standortinformationen werden aus einer Klimadatei bezogen, angegeben im Element `<ClimateFilePath>`. Dieser kann Platzhalter enthalten (siehe Section 2.3).

Zusätzliche Parameter (siehe Section 2.2.1 für eine Beschreibung des Elementtyps `IBK:Parameter`):

Name	Einheit	Beschreibung	Wertebereich	Verwendung
`Albedo`	---	Wird für die Berechnung der diffusen Sonneneinstrahlung verwendet (siehe Section 2.11.3)	0...1	*erforderlich*
(*)`Altitude`	m	wird für bestimmte höhenbezogene Parameter benötigt	>0	*optional*
`Longitude`	Deg	Wenn angegeben, wird der Ortsparameter `Longitude` der Klimadatendatei überschrieben (siehe Section 2.11.2.2).	-180...180	*optional*
`Latitude`	Deg	Wenn angegeben, wird der Ortsparameter `Latitude` der Klimadatendatei überschrieben (siehe Section 2.11.2.2).	-90...90	*optional*

(*) wird noch nicht verwendet.

Flags und Optionen (siehe Section 2.2.3 für eine Beschreibung des Elementtyps `IBK:Flag`):

Name	Beschreibung	Standard	Verwendung
`PerezDiffuseRadiationModel`	Legt fest, ob das Perez-Modell für die Berechnung der diffusen Sonnenstrahlung verwendet werden soll	*false*	*optional*
`ContinuousShadingFactorData`	Wenn gesetzt werden Verschattungsdaten als kontinuierliche Zeitreihen behandelt (siehe Section 2.11.4)	*false*	*optional*

Klimadateien

Derzeit werden `c6b`, `wac` und `epw` Dateien unterstützt (siehe auch Dokumentation zur Software CCM-Editor).

Sie müssen den Pfad zur Klimadatei im `<ClimateFileName>`-Element angeben. Dabei können Sie einen absoluten oder relativen Pfad angeben.

Wenn ein relativer Pfad angegeben wird, wird dieser mit dem aktuellen Arbeitsverzeichnis als Referenz aufgelöst. Wenn Sie zum Beispiel angegeben haben

```
<ClimateFilePath>GER_Potsdam_2017.c6b</ClimateFilePath>
```

und der Solver aus dem Verzeichnis `/home/user/sim/Project1` gestartet wird, wird die Klimadatendatei in `/home/user/sim/Project1/GER_Potsdam_2017.c6b` gesucht. Wenn der Solver von einem anderen Verzeichnis aus gestartet wird, wird die referenzierte Klimadatendatei nicht gefunden und es wird eine Fehlermeldung ausgegeben.

Um dieses Problem zu vermeiden, können Sie Verzeichnisplatzhalter angeben, um den Pfad zur die Klimadatendatei *relativ* zum Speicherort der Projektdatei anzugeben. Der eingebaute Pfadplatzhalter `${Project Directory}` wird durch das Verzeichnis ersetzt, in dem sich die Projektdatei befindet. Verwenden Sie den Platzhalter einfach wie einen regulären Verzeichnisteil, z. B:

```
<ClimateFilePath>${Project Directory}/climate /GER_Potsdam_2017.c6b</ClimateFilePath>
```

Es ist möglich, für alle extern referenzierten Dateien eigene Platzhalter im Projekt zu definieren, siehe Section 2.3.

Gebäude-/Klimastandort

Klimadatendateien enthalten Informationen über Breiten- und Längengrad der Wetterstation, die auch als Standort des Gebäudes angenommen wird. Dadurch wird sichergestellt, dass Simulationszeit und Sonnenstand übereinstimmen.

Es ist jedoch auch möglich, den Breitengrad/Längengrad in der Projektdatei anders zu definieren. Wenn diese Parameter in der Projektdatei angegeben werden (es müssen immer **beide** Parameter angegeben werden und gültig sein), werden diese Parameter aus der Projektdatei anstelle der Standortparameter der Klimadatei verwendet.

Durch die Angabe eines von der Klimastation abweichenden Breitengrades kann der berechnete Sonnenstand nicht mehr mit dem Sonnenstand an der Wetterstation übereinstimmen, was zu möglicherweise falschen Solarstrahlungslasten führt.

Gültiger Wertebereich für `Latitude` ist [-90,90] Grad (positive Werte entsprechen der nördlichen Hemisphäre), für `Longitude` ist es [-180,180] Grad (positive Werte sind östlich von Greenwich).

Zyklische (jährliche) und kontinuierliche (mehrjährige) Klimadaten

Die Klimadaten-Datei kann 8760 Stundenwerte für ein ganzes Jahr enthalten. Anderfalls werden die Klimadaten als kontinuierlich abgelegte Daten für Zeitwerte in einem beliebigen Zeitbereich betrachtet. Dabei können die Klimadaten auch mit variierenden Zeitabständen zwischen den Datenpunkten definiert sein. Solche Klimadateien können nicht für die jährliche/zyklische Berechnung verwendet werden, sondern benötigen ein bestimmtes (passendes) Simulationszeitintervall (siehe Section 2.15.1.1).

Zyklisches Jahresklima

Hierbei werden die Klimadaten in Stundenwerten bereitgestellt. Die Interpretation dieser Werte hängt von der Art der physikalischen Größe ab. NANDRAD unterscheidet zwischen *Zustandsgrößen* und *Fluss-/Lastgrößen*.

Zustandsgrößen sind:

- Temperaturen
- relative Luftfeuchten
- Luftdrücke
- Windrichtung
- Windgeschwindigkeit

Fluss-/Lastgrößen sind:

- direkte Sonnenstrahlungsintensität (in Normalrichtung der Sonne)
- diffuse solare Strahlungsintensität (in horizontaler Ebene)
- Regenlast
- langwellige Himmelsemission/-gegenstrahlung

Es wird erwartet, dass die Zustandsgrößen als *Momentanwerte* am *Ende jeder Stunde* angegeben werden. Substündliche Werte werden durch lineare Interpolation erhalten, wie in Abbildung 3 gezeigt.

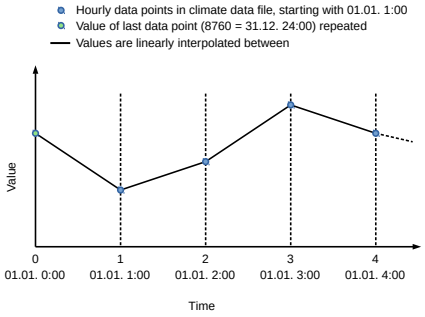


Abbildung 3. Bei Zustandsgrößen wird eine lineare Interpolation verwendet, um den Zeitverlauf zwischen den stündlichen Momentanwerten zu rekonstruieren

Fluss-/Lastgrößen werden als *Mittelwerte* über die *letzte Stunde* erwartet. Die substündlichen Werte werden durch lineare Interpolation zwischen den in der Mitte jeder Stunde platzierten Mittelwerten erhalten, wie in Abbildung 4 gezeigt.

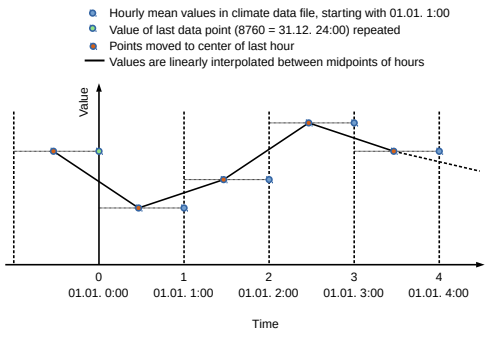


Abbildung 4. Bei Flussgrößen/Lasten werden die Werte in die Stundenmitte verschoben und dann linear interpoliert.

Die so erhaltenen Integralwerte in einer Stunde weichen leicht von den ursprünglichen integralen Mittelwerten ab (siehe z. B. Stunde zwischen 2:00 und 3:00). Die Fehler sind jedoch klein und heben sich im Tagesverlauf fast komplett auf. Dafür sind die generierten Zeitreihen und substündlichen Werte stets stetig.

Kontinuierliche Daten

Hierbei enthält die Klimadatendatei Datenpunkte (mindestens 2), wodurch auch der früheste Start- und späteste Endpunkt der Simulation definiert wird.

Wenn Sie die Simulation über die verfügbaren Klimadaten hinaus fortsetzen, werden die letzten Werte im Klimadatensatz konstant gehalten. Dies führt in der Regel zu sinnlosen Ergebnissen (es sei denn, dies ist in künstlichen Testfällen beabsichtigt).

Da der Benutzer in den Klimadatendateien beliebige Zeitschritte bis hin zu winzigen Werten wählen kann, hängt die Genauigkeit der Eingabedaten von den Benutzereingaben ab. Zwischen den Zeitpunkten wird der Solver **alle Größen** in der Klimadatendatei linear interpolieren und **nicht** wie bei stündlichen Daten zwischen Zuständen und Lasten unterscheiden.

Um das gleiche Ergebnis wie bei jährlichen, zyklischen Stundendaten zu erzielen, müssen Klimadaten in 30-Minuten-Intervallen angegeben und interpolierte Werte am Ende und in der Mitte jeder Stunde selbst berechnet werden.

Zusätzliche Strahlungssensoren

Es ist möglich zusätzliche Ebenen (Sensoren) zu spezifizieren, um Strahlungslasten zu berechnen und für andere Modelle als Sensorgrößen zur Verfügung zu stellen. Dies geschieht durch die Angabe einer `Sensor`-Definition.

Beispiel 26. Definition von Sensoren

```
<Location>
    ....
    <Sensors>
    <!-- Flachdach>
    <Sensor id="1">
      <IBK:Parameter name="Orientation" unit="Deg">0</IBK:Parameter>
      <IBK:Parameter name="Inclination" unit="Deg">0</IBK:Parameter>
    </Sensor>
    <!-- Nordwand 90 -->
    <Sensor id="2">
      <IBK:Parameter name="Orientation" unit="Deg">0</IBK:Parameter>
      <IBK:Parameter name="Inclination" unit="Deg">90</IBK:Parameter>
    </Sensor>
        ...
    </Sensors>
</Location>
```

Tabelle 15. Attribute

Attribut	Beschreibung	Format	Verwendung
`id`	Kennung des Sensors	> 0	*erforderlich*

Parameter (siehe Abschnitt Section 2.2.1 für eine Beschreibung des Elementtyps `IBK:Parameter`):

Name	Einheit	Beschreibung	Wertebereich	Verwendung
`Orientation`	Deg	Ausrichtung des Sensors	0...360	*erforderlich*
`Inclination`	Deg	Neigung des Sensors • 0 Deg - nach oben gerichtet • 90 Grad - z. B. wie eine senkrechte Wand • 180 Grad - nach unten gerichtet	0...180	*erforderlich*

Einem Sensor muss eine eindeutige ID-Nummer und die obligatorischen Parameter `Orientation` und `Inclination` gegeben werden (siehe Section 2.7 für Details zu deren Definition).

Für jeden Sensor werden 4 Ausgangsgrößen erzeugt:

- `DirectSWRadOnPlane[<sensor id>]` - direkte Sonnenstrahlungsintensität auf die Sensorfläche in [W/m2]
- `DiffuseSWRadOnPlane[<sensor id>]` - diffuse Sonnenstrahlungsintensität auf die Sensorfläche in [W/m2]
- `GlobalSWRadOnPlane[<sensor id>]` - globale Strahlungsintensität auf die Sensorfläche in [W/m2] (die Summe der beiden erstgenannten Größen)
- `IncidenceAngleOnPlane[<sensor id>]` - der Sonneneinfallswinkel auf die Sensorfläche in [Grad] (0°, wenn der Sonnenstrahl senkrecht zur Ebene steht, 90°, wenn der Strahl parallel zur Ebene verläuft oder wenn die Sonne unter dem Horizont ist)

Beispiel 27. Ausgabedefinition für Sensorwerte (siehe auch Beschreibung von Ergebnisdefinitionen in Section 2.14).

```
<OutputDefinitionen>
    ...
    <!-- direkte Strahlung intensiv vom Sensor mit id=2 -->
  <OutputDefinition>
    <Quantity>DirektSWRadOnPlane[2]</Quantity>
    <ObjectListName>Location</ObjectListName>
    <GridName>minütlich</GridName>
  </OutputDefinition>
  <!-- Einfallswinkel vom Sensor mit id=42 -->
  <OutputDefinition>
    <Quantity>IncidenceAngleOnPlane[42]</Quantity>
    <ObjectListName>Location</ObjectListName>
    <GridName>minütlich</GridName>
  </OutputDefinition>
    ...
</OutputDefinitions>
```

2.11.3. Sonnenstrahlungsberechnung

Die Berechnung der Sonneneinstrahlung folgt den in der *Physikalischen Modellreferenz* aufgeführten Gleichungen. Der Parameter `Albedo` wird bei der Berechnung der diffusen Strahlungslast verwendet. Die Schalter `PerezDiffuseRadiationModel` beinflusst die Berechnung der Diffusstrahlung.

2.11.4. Vorberechnete externe Verschattung/Eigenverschattung

In einem vorgelagerten Rechenschritt kann für jedes Flächenelement des Gebäudes der Anteil der sonnenbeschienenen Fläche berechnet werden. In Abbildung 5 wird zum Beispiel eine Fassade teilweise verschattet.

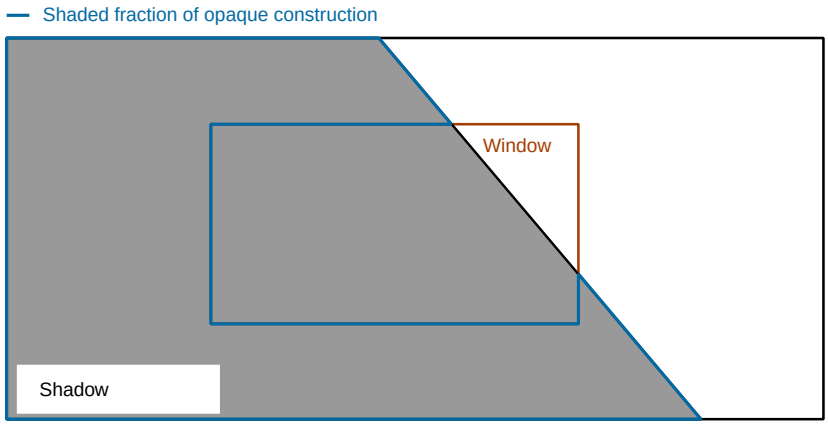


Abbildung 5. Darstellung einer teilverschatteten Fassade mit einem Fenster

Die Software kann nun den Prozentsatz der verschatteten Fläche sowohl für das opake Fassadenelement als auch für das Fensterobjekt separat berechnen. Das Fenster ist zu ca. 80 % verschattet, und ca. 20 % der opaquen Fläche sind noch der Sonne ausgesetzt. Der letztere Anteil wird auch als *Sonnenlichtfaktor* bzw. *Abminderungsfaktor infolge Verschattung* (engl. *shading factors*) bezeichnet.

Der für eine opaque Konstruktion gespeicherte Faktor ist immer *exklusive* aller eingebetteter Objekte zu verstehen. Abbildung 6 zeigt ein ähnliches Beispiel, bei dem die Berechnung der Flächen und Sonnenlichtfaktoren erläutert wird.

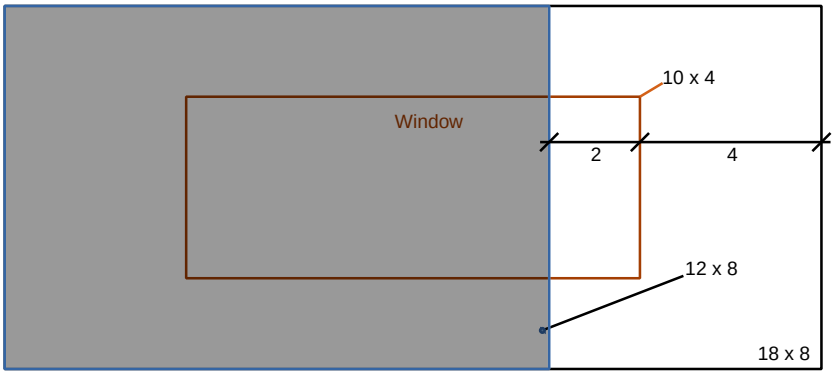


Abbildung 6. Berechnungsbeispiel für eine teilverschattete Fassade mit einem Fenster

Die Konstruktion hat eine Fläche von 18x8 = 144 m2. Das Fenster hat eine Fläche von 10x4 = 40 m2. Damit verbleibt für die eine opaque Konstruktionsfläche von 144 - 40 = 104 m2.

Der Schatten auf dem Fenster allein nimmt 8x4 = 32 m2 ein. Der Sonnenlichtfaktor für das Fenster allein beträgt also 1 - 32/40 = 20%.

Die verschattete Fläche auf der opaken Konstruktion beträgt 12x8 - 8x4 = 96 - 32 = 64 m2. Der Sonnenlichtfaktor, der in der Gleichung für die Belastung durch die Sonneneinstrahlung verwendet werden muss beträgt also 1 - 64/104 = 38,5 %.

Die Werte **0.385** und **0.2** werden in der Datei für die vorberechneten Sonnenlichtfaktoren gespeichert.

Die mittlere Strahlungsintensität auf eine opaque Fläche ergibt sich dann aus:

```
Mittlere direkte Strahlungslast in [W/m2] = Sonnenlichtfaktor (aus Datei) * direkte Strahlungslast
```

Dateiformat für vorberechnete Sonnenlichtfaktoren

Die Datei mit den vorberechneten Sonnenlichtfaktoren wird im XML-Element `Location` definiert, im Kind-Element `ShadingFactorFilePath`. Der hier angegebene Pfad kann ein absoluter Pfad oder ein relativer Pfad sein, der einem Platzhalter folgt (bspw. `${Project Directory}`, siehe Section 2.3).

Die Datei kann als `tsv` Datei oder DataIO-Datei (ASCII oder Binärformat, Endungen `d6o` und `d6b`) bereitgestellt werden.

Die Dateien enthalten für bestimmte, kontinuierlich ansteigende Zeitpunkte die jeweilse berechneten Sonnenlichtfaktoren.

Bei der Berechnung werden die Werte in der Datentabelle linear interpoliert.

Standardmäßig wird von zyklischen Jahresdaten ausgegangen. Dabei müssen die Zeitpunkte stets < 365 d bleiben. Sollen kontinuierliche Daten verwendet werden, muss der Schalter `ContinuousShadingFactorData` eingeschaltet sein.

TSV-Format für Sonnenlichtfaktor-Dateien

Bei Verwendung des `tsv`-Formats müssen die Regeln des `tsv`-Dateiformats (siehe *PostProc 2* Dokumentation) eingehalten werden. Es gibt eine einzelne Kopfzeile. Die erste Spalte ist die Zeitspalte mit Zeit-offsets relativ zu Mitternacht des 1. Januar des Startjahres. Es können beliebige Zeiteinheiten verwendet werden.

Alle anderen Spalten enthalten die berechneten Sonnenlichtfaktoren, wobei jeder Spaltenkopf die jeweilige Fläche mit eindeutiger ID identifiziert. Für opaque Flächen werden die IDs der jeweiligen Konstruktionsinstanzen verwendet. Bei Fenstern werden die IDs der eingebetteten Objekte verwendet. Als Werteeinheit muss `---` verwendet werden.

Beispiel 28. TSV-Datei mit Sonnenlichtfaktoren für 4 Flächen

```
Time [d]  1001 [---]  1002 [---]  1003 [---]  1004 [---]
0 1 1 1 1
181 1 1 1 1
182 0 0 1 1
185 0 0 1 1
186 0 1 1 1
188 0 1 1 1
189 1 1 1 1
```

Fenster/Konstruktion mit IDs 1001 und 1002 werden in den Tagen 182 bis 188 verschattet. Die Fenster 1003 und 1004 bleiben die ganze Zeit unverschattet.

DataIO Format

Bei Verwendung des DataIO-Formats muss das REFERENCE-Format verwendet werden. Das Feld INDICES enthält die IDs der jeweiligen Flächen.

Beispiel 29. DataIO-Datei mit Sonnenlichtfaktoren für 4 Flächen, analog zum obigen TSV-Beispiel

```
D6OARLZ! 007.000
TYPE          = REFERENCE
QUANTITY      = 1001 | 1002 | 1003 | 1004
VALUE_UNIT    = ---
TIME_UNIT     = d
INDICES       = 1001 1002 1003 1004

0 1 1 1 1
181 1 1 1 1
182 0 0 1 1
185 0 0 1 1
186 0 1 1 1
188 0 1 1 1
189 1 1 1 1
```

Für größere Gebäude ist das binäre DataIO-Format (mit gleichem Inhalt) zu empfehlen.

2.12. Objektlisten und Ergebnisreferenzen

Wann immer es notwendig ist, ein Berechnungsergebnis (eines Modellobjekts) zu referenzieren, geschieht dies über die *ObjectLists*.

In NANDRAD werden physikalische Gleichungen in Form von Modellobjekten organisiert, zum Beispiel Zonen oder Konstruktionen. Diese Modellobjekte können durch einen Modelltyp und eine ID-Nummer eindeutig identifiziert werden. Zum Beispiel werden alle für einen Raum/eine Zone berechneten Größen durch den Modelltyp *Zone* und die ID-Nummer der jeweiligen Zone identifiziert. Tabelle 16 listet die verfügbaren Referenztyp-Schlüsselwörter auf.

Tabelle 16. Modell-Referenztypen

Schlüsselwort	Beschreibung
`Zone`	Variablen bezogen auf den Raum (thermische Zonen)
`ConstructionInstance`	Variablen, die sich auf Konstruktionen beziehen
`Schedule`	Geplante Parameter
`Location`	Variablen aus dem Klimaberechnungsmodell, einschließlich Strahlungssensorwerte
`Model`	Modellspezifische Variablen/Ergebnisse

Beispiel 30 zeigt mehrere Beispiele für Definitionen der Objektliste.

Beispiel 30. Definition von mehreren Objektlisten

```
<ObjectLists>
  <ObjectList name="All zones">
    <FilterID>*</FilterID>
    <ReferenceType>Zone</ReferenceType>
  </ObjectList>
  <ObjectList name="Zone Var01">
    <FilterID>1</FilterID>
    <ReferenceType>Zone</ReferenceType>
  </ObjectList>
  <ObjectList name="Wall_1_and_2">
    <FilterID>1,2</FilterID>
    <ReferenceType>ConstructionInstance</ReferenceType>
  </ObjectList>
  <ObjectList name="InfiltrationModel">
    <FilterID>501</FilterID>
    <ReferenceType>Model</ReferenceType>
  </ObjectList>
  ...
</ObjectLists>
```

2.12.1. Objektlisten-Definitionen

Alle Objektlisten werden innerhalb des übergeordneten tags `ObjectLists` definiert. Jede Objektlistendefinition beginnt mit dem XML-tag `ObjectList` mit dem obligatorischen Attribut `name`, das die Objektliste eindeutig identifiziert.

Das XML-tag `ObjectList` hat die folgenden untergeordneten tags.

Tabelle 17. Modell-Referenztypen

Schlüsselwort	Beschreibung
`FilterID`	ID-Filtermuster (siehe Beschreibung unten)
`ReferenceType`	Modellobjekt-Referenztyp (siehe Tabelle 16)

2.12.2. ID-Filter-Muster

Objekte (mit gleichem `ReferenceType`) werden durch ihre ID-Nummer eindeutig identifiziert.

ID-Nummern müssen nur für Objekte mit gleichem `ReferenceType` eindeutig sein. Daher ist es möglich, Zone #1 und ConstructionInstance #1 gleichzeitig zu definieren.

Ein Filtermuster kann aus mehreren Teilen bestehen, die durch , (Komma) getrennt sind, zum Beispiel: `1,4,13-20`. Jeder Teil kann das folgende Format haben:

- eine einzelne ID-Nummer, z. B. *12*
- ein Bereich von ID-Nummern, z. B. *1-100*
- * (wählt alle IDs aus)

Wenn IDs mehrmals angeben werden, z.B. in "3, 1-10", enthält die resultierende ID-Menge jede ID nur einmal.

2.13. Zeitpläne

2.13.1. Übersicht

Zeitpläne liefern rein zeitabhängige Größen, ähnlich wie Klimabelastungen. Im Unterschied zu anderen ergebnisproduzierenden Modellen erzeugen Zeitpläne Variablen für Mengen von abhängigen Modellen. Als solches wird ein Zeitplan für eine Objektliste formuliert, die eine Menge von Objekten mit den vom Zeitplan vorgegebenen Werten auswählt. Sie werden z. B. in den folgenden Objekten oder Objektlisten verwendet:

- **Belegungsraten**, **Wärmelasten**, **Bekleidungsfaktoren** im Personenlastmodell.
- **Heiz-/Kühlsolltemperaturen** für Thermostatsteuerungen
- **Massendurchflussraten** oder **Temperatursollwerte** für Anlagenteile
- **Elektrische Leistungsraten** für Beleuchtung und elektrische Geräte

Ein Zeitplan definiert z. B. einen Heizungssollwert `HeatingSetPoint` für bestimmte Zonen wie z. B. Wohnräume. Diese werden über eine Objektliste mit dem Namen "Living room" ausgewählt, die Objekte vom Typ `Zone` und einem bestimmten ID-Bereich (wird später noch genauer beschrieben) auswählt.

Es gibt zwei Möglichkeiten, einen Zeitplan zu beschreiben:

- `ScheduleGroups`
- `AnnualSchedules`.

Die beiden Möglichkeiten werden in Tagesschema-basierte Zeitpläne und Jahresschaltpläne detailliert besprochen.

Außerdem können Zeitplandaten auf zwei verschiedene Arten behandelt werden, als

- **Zyklische** Daten und
- **Nicht-zyklische** Daten.

Zyklische Daten bedeutet, dass die Zeitplanwerte nach dem Ende der Zeitplanperiode wiederholt werden. Das bedeutet zum Beispiel, dass ein jährlicher Zeitplan zweimal ausgeführt wird, wenn die Simulationszeit auf zwei Jahre eingestellt wird. **Zyklische Daten** können durch `ScheduleGroups` und `AnnualSchedules` definiert werden.

Nicht-zyklische Daten werden immer nur einmal verwendet. Dies ist sinnvoll, wenn gemessene Daten (Monitoring) zum Einrichten von Zeitplänen verwendet werden sollen. Dann muss die Simulation nur für die Zeitspanne eingestellt werden, in der die gemessenen Daten vorhanden sind. **Nicht-zyklische** Daten können nur durch `AnnualSchedules` definiert werden.

Beispiel 31. Definition eines Zeitplans

```
<Schedules>
    <Holidays>5,10</Holidays>
    <WeekEndDays>Fri,Sat</WeekEndDays>
    <FirstDayOfYear>Fri</FirstDayOfYear>
    <IBK:Flag name="EnableCyclicSchedules">true</IBK:Flag>

    <ScheduleGroups>
        ...
    </ScheduleGroups>
    <AnnualSchedules>
        ...
    </AnnualSchedules>
</Schedules>
```

Innerhalb des Objekts `Schedules` können auch die folgenden XML-tags angegeben werden

- `FirstDayOfYear`
- `Holidays`
- `WeekEndDays`
- `Schedule`
- `IBK:Flag` mit dem Namen `EnableCyclicSchedules`

Tabelle 18. mögliche Definitionen von XML-tags

XML-tags	Beschreibung	Format	Verwendung
`FirstDayOfYear`	Der Tagestyp des 1. Januar (Offset des Wochentages des Startjahres. • `Mon` - Monday (*Standard*) • `Tue` - Tuesday • `Wed` - Wednesday • `Thu` - Thursday • `Fri` - Friday • `Sat` - Saturday • `Sun` - Sunday	string	*optional*
`Holidays`	Liste der Feiertage, gespeichert in einer kommagetrennten Liste von Zahlen. Jede Zahl stellt den "Tag des Jahres" dar. Schalttage werden nicht eingeschlossen.	string	*optional*
`WeekEndDays`	Wochenendtage.	string	*optional*
`IBK:Flag`	• Name `EnableCyclicSchedules` ◦ Wenn auf true gesetzt, werden die Zeitpläne nach einem Jahr wiederholt. ◦ Wenn auf false gesetzt (gilt nur für jährliche Zeitpläne), werden diese jährlichen Zeitpläne nur einmal abgetastet.	*true (Standard) / false*	*optional*

2.13.2. Zeitplangruppen

`ScheduleGroup`-tags gibt es sowohl in täglichen schema-basierten Zeitplänen als auch in jährlichen Zeitplänen. Sie identifizieren die Zielobjekte für geplante Parameter. Um Mehrdeutigkeiten zu vermeiden, ist es nicht erlaubt, mehrere Zeitplangruppen mit derselben Objektliste zu definieren.

Definieren Sie nicht mehrere `ScheduleGroup`-Elemente mit der gleichen Objektliste!

2.13.3. Tagesschema-basierte Zeitpläne

Regelmäßige Zeitpläne werden auf Basis eines Tagesschemas definiert. Einige Parameter müssen wie in dem nachfolgenden XML-tag definiert werden.

Beispiel 32. Definition einer Zeitplangruppe

```
<ScheduleGroups>
    <ScheduleGroup objectList="All Zones">
        <!-- AllDays constant -->
        <Schedule type="AllDays">
            ...
        </Schedule>
        <Schedule type="WeekDays">
            ...
        </Schedule>
    </ScheduleGroup>
<ScheduleGroups>
```

Beispiel 33. ObjectList-Definition, die Zonenobjekte auswählt und "All Zones" heißt

```
<ObjectLists>
    <ObjectList name="All Zones">
        <FilterID>*</FilterID>
        <ReferenzTyp>Zone</ReferenzTyp>
    </ObjectList>
</ObjectLists>
```

Regelmäßige Zeitpläne werden innerhalb des XML-tags `ScheduleGroup` mit einem obligatorischen XML-Attribut namens `objectList` definiert, das namentlich auf eine `ObjectList` verweist (siehe Tabelle 19):

Tabelle 19. Attribut für die ScheduleGroup

Name	Beschreibung	Format	Verwendung
`objectList`	Verweise auf eine Objektliste mit dem angegebenen Namen	string	*erforderlich*

Beispiel 32 zeigt eine solche Definition und Beispiel 33 die entsprechende Objektliste.

Tägliche Zyklen

Innerhalb der `ScheduleGroup` können mehrere Objekte namens `Schedule` definiert werden. Die `Schedule`-Objekte benötigen ein XML-Attribut namens `type` mit unterschiedlichen Namen für bestimmte Tagestypen (siehe Tabelle 20). Innerhalb einer `ScheduleGroup` dürfen sich nicht zwei `Schedule`-Objekte mit demselben `type` befinden. Innerhalb jedes `Schedule`-Objekts wird ein Zeitplan definiert, der für alle Tage des angegebenen `Type` im Laufe eines ganzen Jahres gilt. Bei der Konstruktion von Zeitplänen gelten die folgenden Regeln.

In der ersten Priorität werden beim Typ `AllDays` angegebene tägliche Zeitplanwerte (z.B. `HeatingSetPoint`) auf alle Tage des ganzen Jahres gesetzt (Priorität 0). Beispiel 34 zeigt eine solche Zeitplandefinition.

Danach überschreiben die `Types WeekEnd` und `WeekDay`, falls definiert, die bereits definierten Zeitplanwerte nur für alle Wochentage oder Wochenendtage (Priorität 1). Weiterhin definieren die Wochentage namens `Monday`, `Tuesday`, ... für welche Tage die Zeitplanwerte wieder überschrieben werden (Priorität 2). Weiter geht es mit dem Tagestyp `Holiday` (Priorität 3) für die angegebenen Feiertage innerhalb des Objekts `Holidays`.

Es ist möglich, unterschiedliche Zeitpläne für einzelne Zeiträume des Jahres zu definieren, z. B. für das reguläre Jahr und die Sommerferien etc. Auf diese Weise kann ein Zeitplan für das gesamte Jahr definiert werden.

Beispiel 34. Zeitplandefinition mit Typ "AllDays"

```
<ScheduleGroup objectList="Zone01">
    <!-- Konstante "AllDays" -->
    <Schedule type="AllDays">
        <DailyCycles>
            <DailyCycle interpolation="Constant">
                <ZeitPunkte>0</ZeitPunkte>
                <Werte>InfiltrationRateSchedule [1/h]:0</Werte>
            </DailyCycle>
        </DailyCycles>
    </Schedule>
</ScheduleGroup>
```

Tabelle 20 zeigt die Tagestypen und die dazugehörigen Prioritäten.

Tabelle 20. Beschreibung des Attributs "Schedule Type

Type	Priority	Description
`AllDays`	0	Werte werden auf alle Tage der Periode gesetzt
`WeekEnd`	1	Werte werden auf alle Wochenendtage des Zeitraums gesetzt
`WeekDay`	1	Werte werden auf alle Wochentage des Zeitraums gesetzt
`Monday`	2	Werte werden auf alle Montage des Zeitraums gesetzt
`Tuesday`	2	Werte werden auf alle Dienstage des Zeitraums gesetzt
`Wednesday`	2	Werte werden auf alle Mittwoche des Zeitraums gesetzt
`Thursday`	2	Werte werden auf alle Donnerstage des Zeitraums gesetzt
`Friday`	2	Werte werden auf alle Freitage des Zeitraums gesetzt
`Saturday`	2	Werte werden auf alle Samstage des Zeitraums gesetzt
`Sunday`	2	Werte werden auf alle Sonntage des Zeitraums gesetzt
`Holiday`	3	Werte werden auf alle Feiertage des Zeitraums gesetzt, die im tag `holidays` angegeben sind

Beispiel 35 veranschaulicht die Verwendung verschiedener Zeitpläne zur Definition eines Wochenplans. Zunächst wird der grundlegende tägliche Zeitplan definiert. Dann werden spezielle Regeln für Dienstage und Wochenenden definiert. Abbildung 7 veranschaulicht den resultierenden Zeitplan.

Beispiel 35. Zeitplandefinition mit verschiedenen Tagestypen

```
<Schedules>
    <WeekEndDays>Sat,Sun</WeekEndDays>
    <ScheduleGroups>
        <ScheduleGroup objectList="All zones">
            <!-- jeden Tag zwischen 8-10 -->
            <Schedule type="AllDays">
                <DailyCycles>
                    <DailyCycle interpolation="Constant">
                        <TimePoints>0 6 10</TimePoints>
                        <Values>InfiltrationRateSchedule [1/h]:0 0.4 0</Values>
                    </DailyCycle>
                </DailyCycles>
            </Schedule>
            <!-- Dienstag keine Lüftung -->
            <Schedule type="Tuesday">
                <DailyCycles>
                    <DailyCycle interpolation="Constant">
                        <TimePoints>0</TimePoints>
                        <Values>InfiltrationRateSchedule [1/h]:0</Values>
                    </DailyCycle>
                </DailyCycles>
            </Schedule>
            <!-- Wochenende nur am Nachmittag -->
            <Schedule type="WeekEnd">
                <DailyCycles>
                    <DailyCycle interpolation="Constant">
                        <TimePoints>0 14 16</TimePoints>
                        <Values>InfiltrationRateSchedule [1/h]:0 0.1 0</Values>
                    </DailyCycle>
                </DailyCycles>
            </Schedule>
        </ScheduleGroup>
    </ScheduleGroups>
</Schedules>
```

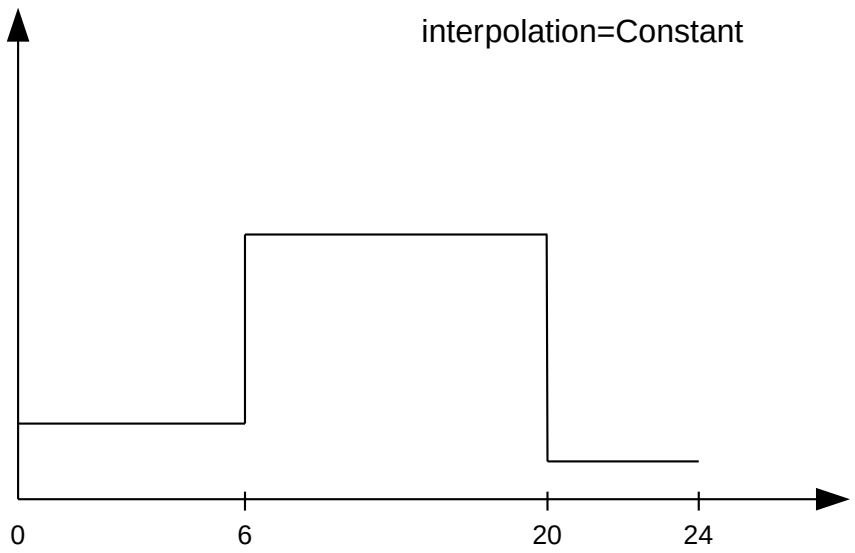


Abbildung 7. Abbildung des wöchentlichen Zeitplans, definiert durch das Beispiel Beispiel 35

DailyCycle Zeitintervalle

Ein `DailyCycle` definiert, wie sich eine oder mehrere Größen im Laufe des Tages ändern. Das untergeordnete tag `TimePoints` definiert durch Leerzeichen getrennte Zeitpunkte in Stunden [h] und damit die verschiedenen Zeitintervalle des Tages.

Wenn das Attribut `interpolation Constant` ist, dann gelten die folgenden Regeln:

- die Zeitpunkte werden als **Startzeit** des nächsten Intervalls interpretiert
- der erste Zeitpunkt muss immer 0 sein, der letzte muss < 24 h sein,
- der entsprechende Wert wird während dieses Intervalls als konstant angenommen

Ein Zeitpunktvektor "0 6 20" definiert z. B. drei Intervalle: 0-6, 6-20, 20-24 und die Datentabelle muss genau 3 Werte enthalten.

Wenn das Attribut `interpolation Linear` ist, dann gelten die folgenden Regeln:

- die Zeitpunkte sind Punkte in der Zeit, an denen zugehörige Werte gegeben sind
- der erste Zeitpunkt muss immer 0 sein, der letzte muss < 24 h sein, denn im zyklischen Betrieb ist der Zeitpunkt bei 24 h derselbe wie bei 0 h (und damit auch die geplanten Werte)
- zwischen den Zeitpunkten werden die Werte linear interpoliert

Abbildung 8 und Abbildung 8 zeigen den resultierenden Werteverlauf für die Zeitintervalle *0, 6, 20* und die entsprechenden Parameterwerte *2, 7, 1*.

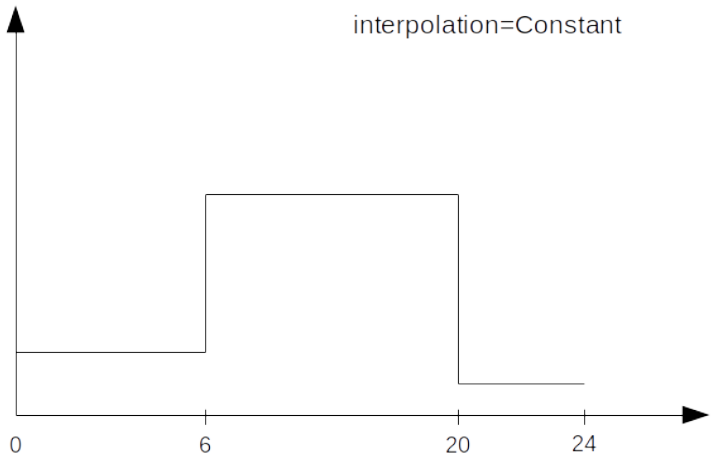


Abbildung 8. Tageszyklus mit konstantem Interpolationsmodus

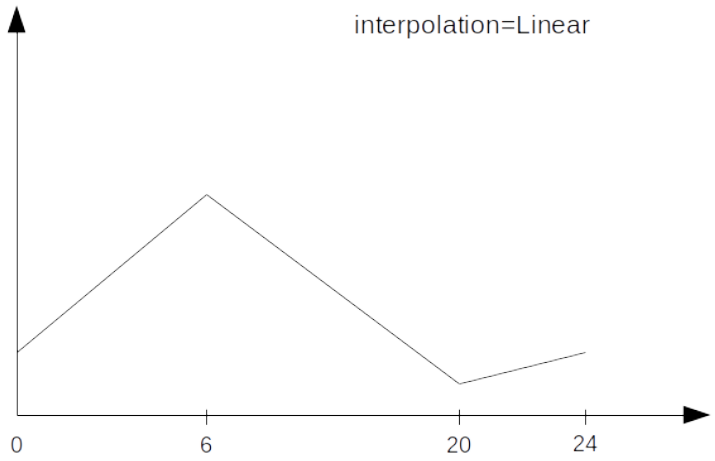


Abbildung 9. Tageszyklus mit linearem Interpolationsmodus

Bei der Verwendung des linearen Interpolationsmodus wird der Wert um 24 Uhr vom Beginn des nächsten Tageszyklus genommen, der im Zeitplan definiert ist. Zum Beispiel würde in Abbildung 7 der Wert am Montag 24:00 Uhr aus dem Zeitplan für Dienstag genommen werden, während der Wert am Mittwoch 24:00 Uhr aus dem regulären Zeitplan *AllDays* genommen würde.

Um ein einzelnes Intervall für den ganzen Tag zu definieren, geben Sie einfach "0" als Wert im XML-tag `TimePoints` an.

Tägliche Zyklusparameterwerte

Für jedes im tag `TimePoints` angegebene Intervall können eine oder mehrere Größen mit zugehörigen Einheiten angegeben werden. Dies geschieht durch die Definition der Datentabelle im XML-Tochtertag `Values` des `DailyCycle` -tags. Die Daten der Datentabelle werden wie folgt formatiert:

```
quantity1 [unit]:val11 val12 val13; quantity2 [unit]:val21 val22 val23;...
```

Grundsätzlich wird jede physikalische Größe in einem string kodiert, wobei die strings für verschiedene Größen zu einem string mit ; (Semikolon) als Trennzeichen zusammengefasst werden.

Jeder Mengenstring setzt sich aus einem Header und den eigentlichen Werten zusammen. Die Werte sind einfach durch Leerzeichen/Tabs oder Komma getrennte Werte (Dezimalzahlen werden mit . (Punkt) als Dezimaltrennzeichen geschrieben).

Der Header ist ein Mengenstichwort (siehe auch Variablenliste), gefolgt von seiner Einheit in Klammern. So hat z. B. eine Heizungssolltemperatur die Kopfzeile `HeatingSetPointTemperature [C]` und die Werte werden dann in Grad C angegeben.

Es müssen *exakt* so viele Werte angegeben werden, wie es Zeitpunkte im XML-tag `TimePoints` gibt. In dieser Datentabelle können Sie so viele Größen angeben, wie Sie benötigen.

Beispiel 36 zeigt einen Tageszyklus mit zwei geplanten Mengen und drei Intervallen.

Beispiel 36. Tageszyklus mit zwei disponierten Mengen

```
<DailyCycle interpolation="Constant">
    <TimePoints>0 6 10</TimePoints>
    <Values>
        InfiltrationRateSchedule [1/h]:0 0.4 0;
        HeatingSetPointTemperature [C]:18 22 18
    </Values>
</DailyCycle>
```

Vermeidung von Sprüngen / Leistungsverbesserung

Bei der Definition von Tageszyklen mit dem Interpolationsmodus `Constant` springen die Werte tatsächlich zwischen den Intervallen. Diese Diskontinuitäten sind sehr teuer in der Berechnung, da der Solver Zeitschritte um diese Sprünge herum gruppieren muss, um den Schrittfunktionen genau zu folgen.

Für praktische Anwendungen sind diese Schritte jedoch oft nicht erwünscht - auch wenn ein Sollwert kurzzeitig auf einen neuen Wert umgeschaltet wird, kann es in der Tat einige Minuten dauern, bis der resultierende physikalische Effekt spürbar wird. Dies wird bei der Interpretation der Sollwerte durch den Solver berücksichtigt.

Anstatt exakt die schrittweise geplanten Werte zu liefern, implementiert der Solver eine automatische 2-Minuten-Rampe kurz vor dem Intervallende. Abbildung 10 veranschaulicht die 2-minütige lineare Rampe, die direkt vor jedem neuen Intervall angewendet wird.

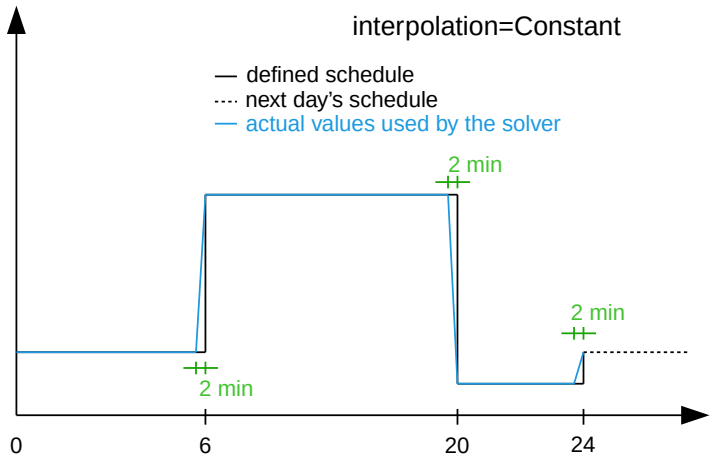


Abbildung 10. Rampen-/Schrittglättung angewandt auf Tageszyklen mit schrittweise definierten Werten

Der Rampenzeitabstand von 2 Minuten ist derzeit in der Zeitplan-Berechnungsroutine fest codiert und kann bei Bedarf auf einen größeren oder kleineren Wert geändert werden. Außerdem kann anstelle einer linearen Rampenfunktion eine polynomische Kurve 3. Ordnung verwendet werden (was immer der bester Kompromiss zwischen Leistung und Genauigkeit ist).

Intern wird die Schrittglättung realisiert, indem 2 Minuten vor dem Intervallende ein neuer Datenpunkt mit dem gleichen Wert wie im aktuellen Intervall eingefügt wird. Der Tageszyklus wird dann wie ein linear interpolierter Tageszyklus behandelt. Es gibt jedoch keine Prüfung für Intervalllängen kleiner als 2 Minuten. Daher **müssen** bei der Definition von Tageszyklen mit Interpolationsmodus `Linear` keine Intervalle kleiner oder gleich 2 Minuten definiert werden.

2.13.4. Jahresschaltpläne

Jahrespläne sind im Grunde Datentabellen mit monoton ansteigenden X (Zeit)-Werten. Jeder Jahresplan definiert eine einzelne Größe. Es können z. B. stündliche Werte von Temperaturen oder Steuergrößen angegeben werden, die während des Jahres gemessen werden.

Der Name *annual schedule* ist eigentlich etwas irreführend. In diesen Datentabellen können Sie Daten mit beliebigen Zeitspannen unterbringen, die nur wenige Wochen oder sogar mehrere Jahre umfassen (z. B. mit Überwachungsdaten). Die einzige Voraussetzung ist, dass das Zeitintervall der Simulation in die Zeitspanne des Zeitplans passt.

Die vom linearen Spline gelieferten Werte können als linear/konstant interpolierte Werte definiert werden, allerdings sollte aus Performance-Gründen der konstante Interpolationsmodus vermieden werden.

Bei linearen Splines wird die Schrittglättung **nicht** vom Solver angewendet. Es ist Sache des Anwenders, geeignete Daten bereitzustellen oder durch langsame Simulationszeiten bestraft zu werden.

Innerhalb des XML-tags `AnnualSchedules` gibt es ein oder mehrere XML-Tochtertags `ScheduleGroup`, jedes mit einem obligatorischen XML-Attribut `objectList`. Dieses referenziert, genau wie bei den täglichen Zyklusplänen eine Objektliste und damit Objekte, auf die sich die geplanten Variablen beziehen. Beispiel 37 zeigt ein Beispiel für jährliche Zeitpläne, die innerhalb einer einzigen `ScheduleGroup` definiert sind.

Beispiel 37. Definition von Jahresplänen

```
<AnnualSchedules>
    ...
    <ScheduleGroup objectList="All zones">
        <AnnualSchedule name="HeatingSetPointTemperature" interpolation="linear">
            <X unit="h"> 0 2183 2184 6576 6577 8760 </X>
            <Y unit="C"> 20 30 20 30 20 30 20 30 </Y>
        </AnnualSchedule>
        <AnnualSchedule name="TotalEnergyProductionPerPerson" interpolation="linear">
            <X unit="h"> 0 2183 2184 6576 6577 8760 </X>
            <Y unit="W/Person"> 70 110 70 110 70 110 </Y>
        </AnnualSchedule>
        <AnnualSchedule name="EquipmentUtilizationRatio" interpolation="linear">
            <X unit="h"> 0 2183 2184 6576 6577 8760</X>
            <Y unit="W/Person"> 10 20 10 20 10 20 </Y>
        </AnnualSchedule>
    </ScheduleGroup>
    ...
</AnnualSchedules>
```

Die eigentlichen Daten werden in den XML-tags `AnnualSchedule` angegeben, die eigentlich ein LinearSplineParameter

sind (siehe referenzierte Dokumentation für Details).

Die Einheit des X-Wertes muss eine Zeiteinheit sein. Die Einheit des Y-Wertes ist die Einheit der geplanten Menge.

2.13.5. Variablenliste

Die Variablenliste beschreibt alle Namen und die Einheiten, die in den Zeitplänen verwendet werden können.

Tabelle 21. Variablenliste

Name	Einheit	Beschreibung
HeatingSetPointTemperature	C	Sollwerttemperatur für Heizung
CoolingSetPointTemperature	C	Sollwerttemperatur für Kühlen
AirConditionSetPointTemperature	C	Solltemperatur für die Klimatisierung
AirConditionSetPointRelativeHumidity	%	Sollwert der relativen Luftfeuchtigkeit für die Klimatisierung
AirConditionSetPointMassFlux	kg/s	Sollwert Massenstrom für die Klimatisierung
HeatingLoad	W	Heizlast
ThermalLoad	W	Thermische Last (positiv oder negativ)
MoistureLoad	g/h	Feuchtelast
CoolingPower	W	Kühlleistung
LightingPower	W	Beleuchtungsleistung
DomesticWaterSetpointTemperature	C	Solltemperatur für Brauchwasser
DomesticWaterMassFlow	kg/s	Massenstrom des Brauchwasserbedarfs für die gesamte Zone (Warmwasser und Geräte)
ThermalEnergyLossPerPerson	W/Person	Energie der Aktivitäten einer einzelnen Person, die nicht als Heizwärme zur Verfügung steht
TotalEnergyProductionPerPerson	W/Person	Gesamtenergieproduktion des Körpers einer einzelnen Person bei einer bestimmten Tätigkeit
MoistureReleasePerPerson	kg/s	Feuchtigkeitsabgabe eines einzelnen Personenkörpers bei einer bestimmten Tätigkeit
CO2EmissionPerPerson	kg/s	CO2-Emissionsmassenstrom einer einzelnen Person bei einer bestimmten Tätigkeit
MassFluxRate	---	Fraktion des realen Massenstroms zum maximalen Massenstrom für verschiedene Tageszeiten
PressureHead	Pa	Versorgungsdruckhöhe einer Pumpe
OccupancyRate	---	Fraktion der realen Belegung zur maximalen Belegung für verschiedene Tageszeiten
EquipmentUtilizationRatio	---	Verhältnis des Verbrauchs für vorhandene elektrische Geräte
LightingUtilizationRatio	---	Verhältnis des Verbrauchs für die Beleuchtung
MaximumSolarRadiationIntensity	W/m2	Maximale Sonneneinstrahlungsintensität bevor die Beschattung aktiviert wird
UserVentilationAirChangeRate	1/h	Austauschrate für natürliche Lüftung

Name	Einheit	Beschreibung
UserVentilationComfortAirChangeRate	1/h	Maximale Luftwechselrate = Offset für Nutzerkomfort
UserVentilationMinimumRoomTemperature	C	Temperaturgrenze, ab der die Komfortlüftung aktiviert wird
UserVentilationMaximumRoomTemperature	C	Temperaturgrenzwert, unterhalb dessen die Komfortlüftung aktiviert wird
InfiltrationAirChangeRate	1/h	Austauschrate für Infiltration
ShadingFactor	---	Schattierungsfaktor [0...1]

2.14. Outputs/Ergebnisse

In NANDRAD ist es möglich, Ausgabedaten für jede berechnete und veröffentlichte Größe abzurufen (siehe Mengenreferenzen für eine vollständige Liste). Natürlich sind nicht alle Größen in allen Projekten verfügbar - vieles hängt davon ab, welche Art von Modellen und Geometrien definiert werden.

Um eine Ausgabe zu definieren, werden die folgenden Informationen benötigt:

- ein Ausgaberaster, definiert *wann* Ausgaben geschrieben werden sollen
- den Variablennamen (Bezeichner für die physikalische Größe)
- eine Objektliste, die das Objekt oder die Objekte auswählt, von denen Daten abgerufen werden sollen
- (optional) Informationen zur Zeitbehandlung, d. h. ob ein zeitlicher Mittelwert gebildet oder eine Zeitintegration durchgeführt werden soll
- (optional) Zieldateiname

Zusätzlich zu den manuell definierten Ausgaben erzeugt NANDRAD auch automatisch eine Reihe von Log- und Datendateien (siehe Abschnitt Solver-Logdateien).

Die Ausgaben werden im XML-tag Outputs definiert, mit der folgenden allgemeinen Struktur:

Beispiel 38. Parameter-Definition für Ausgaben

```
<Outputs>
    ... <!-- globale Ausgabeparameter -->

    <Grids>
        ... <!-- Definition von Ausgaberastern -->
    </Grids>

    <Definitions>
        ... <!-- Tatsächliche Ausgangsdefinitionen -->
    </Definitions>
</Outputs>
```

2.14.1. Globale Ausgabeparameter

Die folgenden Parameter beeinflussen die Erzeugung der Ausgabedateien:

- TimeUnit - der Wert dieses XML-tags enthält die Zeiteinheit, die in den Ausgabedateien verwendet werden soll

(nur bei Dateien im ASCII-Format)

- `IBK:Flag` - namens `BinaryFormat`: falls wahr, werden die Dateien im Binärformat geschrieben (siehe Binäres Format).

Beispiel 39. Globale Ausgabeparameter

```
<Outputs>
    <TimeUnit>d</TimeUnit>
    <IBK:Flag name="BinaryFormat">false</IBK:Flag>
    ....
</Outputs>
```

2.14.2. Ausgaberaster

Ausgaberaster legen fest, *wann* Ausgaben geschrieben werden. Ein Ausgaberaster enthält eine Liste von Intervallen, wobei für jedes Intervall eine Ausgabenschrittgröße definiert ist. Wenn Sie z. B. stündliche Ausgabeschritte von Anfang bis Ende haben möchten, müssen Sie ein Raster mit einem Intervall und einem Schrittgrößenparameter von einer Stunde definieren:

Beispiel 40. Ausgaberaster für die gesamte Simulation mit stündlichen Schritten

```
<Grids>
  <OutputGrid name="hourly">
    <Intervals>
      <Interval>
        <IBK:Parameter name="StepSize" unit="h">1</IBK:Parameter>
      </Interval>
    </Intervals>
  </OutputGrid>
</Grids>
```

Ein Ausgaberaster wird durch seinen Namen (obligatorisches XML-Attribut `name`) eindeutig identifiziert. Es enthält ein einzelnes untergeordnetes XML-tag `Intervals`, das ein oder mehrere Intervalle enthält. Es wird erwartet, dass die Intervalle (XML-tag `Interval`) zeitlich aufeinander folgen. Es sind Lücken dazwischen möglich.

Name	Einheit	Beschreibung	Werteber eich	Verwendung
`Start`	h	die Startzeit des Intervalls (siehe Erklärung unten) der Wand	>=0.0	*optional/erforderlich*
`End`	h	die Endzeit des Intervalls (siehe Erläuterung unten)	>=0.0	*optional/erforderlich*
`StepSize`	h	der Abstand zwischen den Ausgängen innerhalb des Intervalls	>0.0	*erforderlich*

Die Parameter werden in XML-tags vom Typ `IBK:Parameter` gespeichert, siehe IBK:Parameter.

Die Zeitpunkte in den Parametern `Start` und `End` werden in Bezug auf Mitternacht des 1. Januar des jeweiligen

Jahres definiert, in dem die Simulation beginnt.

Regeln

- der Parameter `Start` ist unter den folgenden Bedingungen optional:
  - im ersten Intervall wird ein fehlender `Start`-Parameter automatisch auf 0 gesetzt (Beginn des Jahres)
  - in allen anderen Intervallen wird die `End`-Zeit des vorangegangenen Intervalls genommen (siehe folgende Regel)
- die Endzeit eines Intervalls wird definiert, entweder:
  - durch Definition des Parameters `End`,
  - durch Definition des Parameters `Start` im nächsten Intervall
  - durch die Endzeit der Simulation (nur im letzten Intervall)

Grundsätzlich muss für den Solver klar sein, wann ein Intervall beginnt und endet, und wie groß die Schrittweite ist.

Während der Simulation wird eine Ausgabe genau unter der folgenden Bedingung geschrieben:

- t muss innerhalb eines durch das Gitter definierten Intervalls liegen
- der Offset t vom Beginn des Intervalls muss ein exaktes Vielfaches der Schrittweite sein

Beispiel 41. Ausgaberasterauswertung

Angenommen, ein Ausgabeintervall ist so definiert, dass es bei 12,5 h beginnt, mit einer Schrittweite von 2 h. Die Simulationszeit soll t=16,5 h betragen. Dann wäre 16,5 - 12,5 = 4 h. 4h ist ein exaktes Vielfaches von 2 h. Das Ausgaberaster wäre also zu diesem Simulationszeitpunkt "aktiv" und alle Ausgaben, die mit diesem Ausgangsgitter verbunden sind, werden geschrieben.

Zwischen den Intervallen kann es Lücken geben, in denen keine Ausgaben geschrieben werden:

Beispiel 42. Ausgaberaster für Tageswerte im ersten Jahr und Stundenwerte im dritten Jahr (beginnend zum Zeitpunkt "2 a")

```
<Grids>
  <OutputGrid name="first_and_last">
    <Intervals>
      <Interval>
        <IBK:Parameter name="StepSize" unit="d">1</IBK:Parameter>
        <IBK:Parameter name="End" unit="a">1</IBK:Parameter>
      </Interval>
      <Interval>
        <IBK:Parameter name="Start" unit="a">2</IBK:Parameter>
        <IBK:Parameter name="StepSize" unit="h">1</IBK:Parameter>
      </Interval>
    </Intervals>
  </OutputGrid>
</Grids>
```

2.14.3. Ausgangsdefinitionen

Nachfolgend finden Sie ein Beispiel für eine Ausgabedefinition:

Beispiel 43. Ausgabe der Lufttemperatur von allen Zonen in der Objektliste All zones und unter Verwendung des Ausgaberasters hourly

```
<Definitions>
    <OutputDefinition>
      <Quantity>AirTemperature</Quantity>
      <ObjectListName>All zones</ObjectListName>
      <GridName>hourly</GridName>
    </OutputDefinition>
    ... <!-- weitere Definitionen -->
</Definitions>
```

Das Beispiel zeigt die obligatorischen Elemente des XML-tags `OutputDefinition`. Im Folgenden finden Sie eine Liste aller unterstützten Elemente:

XML-tag	Beschreibung	Verwendung
`Quantity`	Eindeutiger ID-Name der physikalischen Größe, siehe auch Mengenreferenzen	*erforderlich*
`ObjectListName`	Referenz auf eine Objektliste, die die Objekte identifiziert von denen Ergebnisse genommen werden sollen	*erforderlich*
`GridName`	Referenz auf ein Ausgaberaster (Ausgabezeitdefinitionen)	*erforderlich*
`FileName`	Zieldateiname	*optional*
`TimeType`	Methode der Zeitmittelung/Integration	*optional*

Der ID-Name der Ergebnisgröße ist der Name des Ergebnisses eines Modellobjekts, eines Zeitplans oder eines anderen vom Solver erzeugten Objekts. Das entsprechende Objekt oder die entsprechenden Objekte werden durch eine Objektliste ausgewählt. Der Gittername ist der ID-Name eines Ausgaberasters.

Das Element `FileName` ist optional. Er kann verwendet werden, um gezielt den Namen einer Ausgabedatei auszuwählen. Normalerweise werden die Namen der Ausgabedateien automatisch generiert, abhängig von der Art der angeforderten Ausgabe.

Schließlich kann das Element `TimeType` verwendet werden, um die zeitliche Mittelung oder die zeitliche Integration von Variablen festzulegen, siehe Abschnitt Zeittypen.

Variablennamen und Variablennachschlagregeln

Mengen in Ausgabedefinitionen definieren die ID-Namen der Ausgabegrößen. Wenn ein Element einer vektoriellen Größe angefordert wird, muss das betreffende Element über eine Index-Notation definiert werden. Dabei sind die folgenden Notationen erlaubt:

- `HeatSource[1]` - das Index-Argument wird so interpretiert, wie es von den bereitstellenden Modellen definiert wird, wenn also das Modell eine vektorwertige Größe mit Modell-ID-Indizierung bereitstellt, wird das Argument als Objekt-ID interpretiert (ansonsten als Positionsindex)
- `HeatSource[index=1]` - das Argument index wird explizit als Positionsindex interpretiert (führt zu einem Fehler,

wenn das Modell eine Größe mit Modell-ID-Indizierung bereitstellt)

- `HeatSource[id=1]` - das index-Argument wird explizit als Objekt-ID interpretiert (führt zu einem Fehler, wenn das Modell eine Menge mit Positionsindizierung liefert)

Ausgabedateinamen

Die folgenden Abschnitte beschreiben die Regeln, die die Ausgabedateinamen bestimmen.

Wenn kein Dateiname angegeben wird

Zieldateiname(n) werden automatisch festgelegt.

Alle Ausgaben werden abhängig von der physikalischen Größe gruppiert in:

- Zustände : *states*
- Ströme : *fluxes*
- Lasten : *load*
- Sonstiges : *misc*

Wenn `Integral` als `TimeType` gewählt wird:

- für Ausgaben vom Typ *fluxes* wird stattdessen die Gruppe *flux_integrals* verwendet,
- für Ausgaben vom Typ *loads* wird stattdessen die Gruppe *load_integrals* verwendet

Die Ausgaben werden weiter nach dem Namen des Ausgaberasters gruppiert. Der endgültige Ausgabedateiname wird für jeden Gitter- und Gruppennamen ermittelt:

- *states* → `states_<gridname>.tsv`
- *loads* → `loads_<gridname>.tsv`
- *loads (integriert)* → `load_integrals_<gridname>.tsv`
- *fluxes* → `fluxes_<gridname>.tsv`
- *fluxes (integriert)* → `flux_integrals_<gridname>.tsv`

Es gibt eine Sonderregel: Wenn nur ein Gitter verwendet wird, wird das Suffix `_<gridname>` weggelassen.

Wenn ein Dateiname angegeben wird

Die Menge wird in die angegebene Datei geschrieben. Wenn es mehrere Ausgabedefinitionen mit demselben Dateinamen gibt, werden alle Mengen in dieselbe Datei geschrieben, unabhängig vom Typ.

Alle Ausgabedefinitionen mit demselben Dateinamen müssen das **gleiche** Raster verwenden (gleiche Zeitpunkte für alle Spalten sind erforderlich!)

Zeittypen

Das tag `TimeType` nimmt die folgenden Werte an:

- `None` - schreibt die Ausgaben wie zum Ausgabezeitpunkt errechnet
- `Mean`- schreibt den über das letzte Ausgabeintervall gemittelten Wert
- `Integral` - schreibt das Zeitintegral der Ergebnisgröße (Integration beginnt zu Simulationsbeginn stets bei 0)

Standardmäßig (wenn das Element `TimeType` nicht explizit angegeben ist) werden die Werte so geschrieben, wie sie zum Ausgabezeitpunkt berechnet werden (entspricht `None`). Abbildung Illustration der verschiedenen `TimeType` -Optionen veranschaulicht die verschiedenen Optionen.

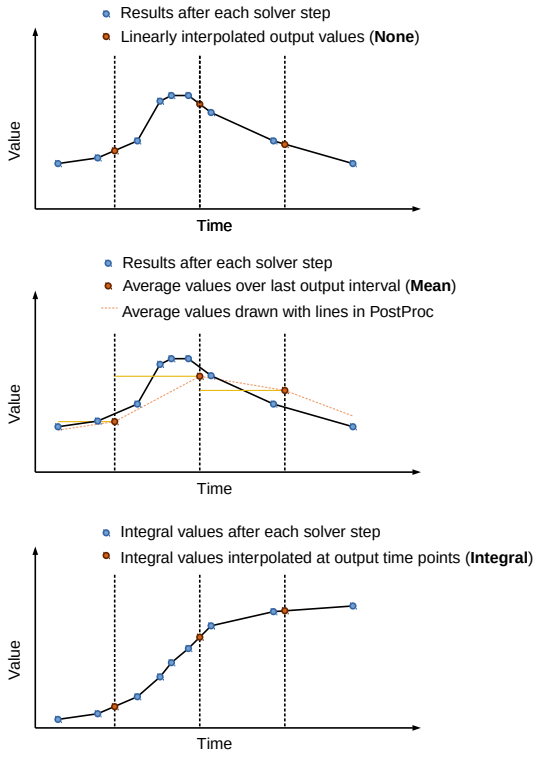


Abbildung 11. Illustration der verschiedenen `TimeType`-Optionen

Es ist wichtig zu beachten, dass Durchschnittswerte immer Mittelwerte der Werte im *letzten Ausgabeintervall* sind. Wenn Sie also stündliche Ausgänge definiert haben, aber die Einheit `kW/d` ist, erhalten Sie keine Durchschnittswerte über einen Tag, sondern über die letzte Stunde. Die Einheit wird nur zur Umrechnung des Endwertes benötigt, hat aber keinen Einfluss auf die Art der Berechnung.

Beispiele

Beispiel 44. Abfrage von Oberflächentemperaturen der Konstruktionen

```
<Outputs>
    ...
    <Definitions>
        <OutputDefinition>
          <Quantity>SurfaceTemperatureA</Quantity>
          <ObjectListName>Walls</ObjectListName>
          <GridName>hourly</GridName>
        </OutputDefinition>
        <OutputDefinition>
          <Quantity>SurfaceTemperatureB</Quantity>
          <ObjectListName>Walls</ObjectListName>
          <GridName>hourly</GridName>
        </OutputDefinition>
        ... <!-- weitere Definitionen -->
    </Definitions>
</Outputs>
<ObjectLists>
  <ObjectList name="Walls">
    <FilterID>*</FilterID>
    <!-- Objektliste muss auf Konstruktionsinstanzen verweisen -->
    <ReferenceType>ConstructionInstance</ReferenceType>
  </ObjectList>
    ... <!-- andere Objektlisten -->
</ObjectLists>
```

Beispiel 45. Anforderung von Energie, die der Schicht in einer Konstruktion zugeführt wird (Fußbodenheizung)

```
<Outputs>
    ...
    <Definitions>
        <OutputDefinition>
            <!-- Index 1 = Wärmequelle in Schicht 1, von Seite A aus zählend -->
          <Quantity>HeatSource[1]</Quantity>
          <ObjectListName>FloorHeating1</ObjectListName>
          <GridName>hourly</GridName>
        </OutputDefinition>
        ... <!-- weitere Definitionen -->
    </Definitions>
</Outputs>
<ObjectLists>
  <ObjectList name="FloorHeating1">
    <FilterID>15</FilterID>
    <!-- Objektliste muss Bauinstanzen referenzieren -->
    <ReferenceType>ConstructionInstance</ReferenceType>
  </ObjectList>
    ... <!-- andere Objektlisten -->
</ObjectLists>
```

2.14.4. Binäres Format

Die binäre Variante von TSV-Dateien ist sehr ähnlich.

```
Kopfsatz:
- 64bit Ganzzahl = n (Anzahl der Spalten)
- n mal binäre Zeichenketten

Datenteil, jeder Datensatz:



Erster Datensatz: unsigned int - n (Anzahl der Spalten)
Nächste n Datensätze: Binärstrings, führende Größe (unsigned int) und Abschlusszeichen (Integritätsprüfung)

Als Nächstes ?? Datensätze: unsigned int - n (zur Überprüfung) und danach n Doubles
```

TODO Andreas? Wenn das erste Mal Ausgabendateien > 100 Mb als ASCII-Dateien geschrieben werden, sollten wir das binäre Format einführen.

2.14.5. Solver-Logdateien

Innerhalb des Ergebnisverzeichnisses des Projekts werden automatisch die folgenden Dateien erzeugt:

```
├──── log
│    ├──── integrator_cvode_stats.tsv
│    ├──── LES_direct_stats.tsv
│    ├──── progress.tsv
│    ├──── screenlog.txt
│    └──── summary.txt
├──── results
│    └──── ... (output files)
└──── var
     ├──── input_reference_list.txt
     ├──── objectref_substitutions.txt
     ├──── output_reference_list.txt
     └──── restart.bin
```

Datei	Beschreibung
integrator_cvode_stats.tsv	Statistik des Zeitintegrators, wird am Ende der Simulation geschrieben
LES_direct_stats.tsv	Statistik des Linear Equation System (LES) Solvers, wird am Ende der Simulation geschrieben
progress.tsv	Minimalistische Laufzeit-Fortschrittsdaten, kontinuierlich geschrieben, kann zum Verfolgen des Simulationsfortschritts vom GUI-Tool verwendet werden
screenlog.txt	Log-Datei für Solver-Ausgabemeldungen (wie Konsolenfensterausgaben), wird kontinuierlich geschrieben
summary.txt	Statistiken und Zeitangaben des Simulationslaufs, wird am Ende der Simulation geschrieben
input_reference_list.txt	Liste der von Modellen verwendete Eingangsgrößen (Ergebnisgrößen anderer Modelle) (siehe Mengenreferenzen)
output_reference_list.txt	Liste der in diesem Projekt erzeugten Größen (siehe Mengenreferenzen)
objectref_substitutions.txt	Liste von Objektreferenzen (einschließlich IDs), wie sie in Ausgabedateien erscheinen und deren *Displayname* Attributen (wenn vergeben). Kann benutzt werden, um die generischen Bezeichner in lesbare Begriffe zu übersetzen.

Datei	Beschreibung
restart.bin	Binäre Neustartdaten (zur Fortsetzung der Integration/des Solvers)

Wenn Sie einen anderen Integrator oder Solver für lineare Gleichungssysteme gewählt haben (siehe Abschnitt Solver-Parameter), werden die Dateien integrator_cvode_stats.tsv und LES_direct_stats.tsv entsprechend anders benannt.

2.15. Globale Parameter

Durch die globalen Simulationsoptionen werden folgende Punkte gesteuert:

- wie das Modell arbeitet
- die Berechnungsgenauigkeit (wirkt sich auf die Leistung aus)
- die Berechnungsleistung

Die einzelnen Einstellungen sind aufgeteilt in *Simulationsparameter* und *Solver-Parameter*, wobei sich letztere auf das numerische Lösungsverfahren beziehen.

2.15.1. Simulationsparameter

Im Folgenden werden alle Simulationsparameter beschrieben, siehe Beispiel 46. Alle Parameter werden als IBK:Parameter, IBK:Flags oder IBK:IntPara gesetzt.

Beispiel 46. Simulationsparameter

```
<SimulationParameter>
  <IBK:Parameter name="InitialTemperature" unit="C">5</IBK:Parameter>
  <IBK:IntPara name="DiscMaxElementsPerLayer">30</IBK:IntPara>
  <Interval>
    <IBK:Parameter name="Start" unit="d">0</IBK:Parameter>
    <IBK:Parameter name="End" unit="d">730</IBK:Parameter>
  </Interval>
</SimulationParameter>
```

Das tag SimulationParameter enthält die folgenden untergeordneten tags:

XML-tag	Beschreibung	Verwendung
IBK:Parameter	Schwebepunktwertparameter	mehrfach
IBK:IntPara	Ganzzahlige Parameter	vielfach
IBK:Flag	Flags	vielfach
Interval	Definiert das Simulationsintervall	kein/einmal

Fließkommaparameter (siehe Abschnitt IBK:Parameter für eine Beschreibung des tags IBK:Parameter):

Name	Vorgabeeinheit	Beschreibung	Wertebereich	Verwendung
InitialTemperature	C	Globale Anfangstemperatur für alle Objekte (Zonen, Bauinstanzen, etc)	positive double (> 0.0 K)	*optional*
(*)InitialRelativeHumidity	%	Globale anfängliche relative Luftfeuchtigkeit für alle Objekte, die einen Feuchtewert gesetzt haben können (Zonen, Luftströme in Modellen, etc)	0 ... 100%	*optional*
(*)RadiationLoadFraction	---	Prozentualer Anteil der solaren Strahlungsgewinne, die direkt dem Raum zugerechnet werden 0..1.	0...1	*optional*
(*)UserThermalRadiationFraction	---	Prozentualer Anteil der Wärme, die durch langwellige Strahlung von Personen abgegeben wird.	0...1	*optional*
(*)EquipmentThermalLossFraction	---	Prozentualer Anteil der Energie aus der Gerätebelastung, der nicht als thermische Wärme zur Verfügung steht.	0 ... 1	*optional*
(*)EquipmentThermalRadiationFraction	---	Prozentualer Anteil der Wärme, die durch langwellige Strahlung von Geräten abgegeben wird.	0...1	*optional*
(*)LightingVisibleRadiationFraction	---	Prozentualer Anteil der Energie der Beleuchtung, die in sichtbare kurzwellige Strahlung umgewandelt wird.	0...1	*optional*
(*)LightingThermalRadiationFraction	---	Prozentualer Anteil der Energie der Beleuchtung, die in langwellige Strahlung umgesetzt wird.	0...1	*optional*
(*)DomesticWaterSensitiveHeatGainFraction	---	Prozentualer Anteil der sensiblen Wärme des Brauchwassers, der an den Raum abgegeben wird.	0...1	*optional*
(*)AirExchangeRateN50	1/h	Luftwechselrate, die sich aus einer Druckdifferenz von 50 Pa zwischen innen und außen ergibt.	positive double (> 0.0)	*optional*
(*)ShieldingCoefficient	---	Abschirmkoeffizient für einen bestimmten Ort und Hüllentyp.	0 ... 1	*optional*
(*)HeatingDesignAmbientTemperature	C	Umgebungstemperatur für einen Auslegungstag. Parameter, der für den FMU-Export benötigt wird.	positive double (> 0.0)	*optional*

(*) - bisher noch nicht verwendet

Ganzzahlige Parameter (siehe Abschnitt IBK:IntPara für eine Beschreibung des tags IBK:IntPara):

Name	Beschreibung	Standard	Verwendung
StartJahr	Startjahr der Simulation	*2001*	*optional*

Flags und Optionen (siehe Abschnitt IBK:Flag für eine Beschreibung des tags IBK:Flag):

Name	Beschreibung	Standard	Verwendung
(*)EnableMoistureBalance	Flag, das die Berechnung der Feuchtigkeitsbilanz aktiviert, wenn diese aktiviert ist	*false*	*optional*
(*)EnableCO2Balance	Flag, das die Berechnung der CO2-Bilanz aktiviert, wenn aktiviert	*false*	*optional*
(*)EnableJointVentilation	Flag, das die Belüftung durch Fugen und Öffnungen aktiviert.	*false*	*optional*
(*)ExportClimateDataFMU	Flag, die den FMU-Export von Klimadaten aktiviert.	*false*	*optional*

(*) - bisher noch nicht verwendet

Simulationszeitintervall

Der tag SimulationParameters enthält auch den Start und das Ende der Simulation. Standardmäßig ist das Simulationszeitintervall so eingestellt, dass es sich über ein ganzes Jahr erstreckt, beginnend um Mitternacht am 1. Januar. Es ist jedoch möglich, ein anderes Zeitintervall zu definieren und damit auch eine Simulation, die länger als ein Jahr läuft.

Dies wird im untergeordneten tag Interval gemacht:

Das Simulationsintervall beginnt am 1. Februar (kurz nachdem die ersten 31 Tage des Januars vorbei sind) und läuft 60 Tage.

```
<Interval>
  <IBK:Parameter name="Start" unit="d">31</IBK:Parameter>
  <IBK:Parameter name="End" unit="d">91</IBK:Parameter>
</Interval>
```

Der Start und das Ende einer Simulation werden immer in *simulation time* definiert, was im nächsten Abschnitt genauer erklärt wird.

Simulationszeit und absoluter Zeitbezug

NANDRAD verwendet zwei Zeitmaße:

- **Simulationszeit**, die beim Start der Simulation immer bei 0 beginnt, und
- **Absolute Zeit**, die die in ein reales Datum/Uhrzeit umgerechnete Zeit ist und auf dem tatsächlichen Startzeitpunkt der Simulation basiert.

Die *Simulationszeit* beschreibt grundsätzlich einen Zeitversatz relativ zum Startpunkt der Simulation und wird

typischerweise nur als Zeitdelta ausgedrückt, z. B. "20 d" oder "15.5 h".

Die *Absolute Zeit* ist eine bestimmte Zeit/ein bestimmtes Datum, z. B. "20.09.2020 14:30", die/der sich durch Addition des Offsets der *Simulationszeit* zu einem Startzeitpunkt ergibt.

In NANDRAD wird dieser Simulationsstartzeitpunkt in zwei Parametern angegeben:

- das `StartYear` und
- das Offset der Zeit seit Beginn (Mitternacht 1. Januar) dieses Jahres als `Start` Intervallparameter.

Ein `Start`-Offset von `1 d` lässt die Simulation am *Januar 2, 0:00* beginnen. Wenn die Simulation z.B. am *15. Januar 2003, 6:00* beginnen soll, muss folgendes angegeben werden:

```
StartYear = 2003
Start = 14*24 + 6 = 342 h
```

Und für den letzten Tag des Jahres muss die Simulation bei `Start = 364 d` gestartet werden.

In NANDRAD gibt es keine Schaltjahre. Selbst wenn Sie 2004 als Startjahr angeben, wird es keinen 29. Februar geben! Wenn Sie eine Mehrjahressimulation durchführen, hat jedes Jahr 365 Tage.

2.15.2. Solver-Parameter

Im Folgenden werden alle Parameter beschrieben, die für den Solver benötigt werden.

Solver-Parameter

```
<SolverParameter>
  <IBK:Parameter name="MaxTimeStep" unit="min">30</IBK:Parameter>
  <IBK:Parameter name="MinTimeStep" unit="s">1e-4</IBK:Parameter>
  <IBK:Parameter name="RelTol" unit="---">1e-005</IBK:Parameter>
  <IBK:Parameter name="AbsTol" unit="---">1e-006</IBK:Parameter>
  <IBK:Parameter name="NonlinSolverConvCoeff" unit="---">1e-05</IBK:Parameter>
  <IBK:IntPara name="MaxKrylovDim">30</IBK:IntPara>
  <IBK:Parameter name="DiscMinDx" unit="mm">2</IBK:Parameter>
  <IBK:Parameter name="DiscStretchFactor" unit="---">4</IBK:Parameter>
  <IBK:Flag name="DetectMaxTimeStep">true</IBK:Flag>
  <Integrator>CVODE</Integrator>
  <LesSolver>Dense</LesSolver>
</SolverParameter>
```

Der tag `SolverParameter` enthält die folgenden untergeordneten tags:

XML-tag	Beschreibung	Verwendung
`IBK:Parameter`	Parameter für Fließkommazahlen	mehrfach
`IBK:IntPara`	Ganzzahlige Parameter	vielfach

XML-tag	Beschreibung	Verwendung
IBK:Flag	Flags	mehrfach
Integrator	Definiert Zeitintegrator	kein/einmal
LesSolver	Definiert Solver für lineare Gleichungssysteme (LES)	kein/einmal
Preconditioner	Definiert Vorkonditionierer (nur iterativer LES-Solver)	einzeln/einmal

Fließkommaparameter (siehe Abschnitt IBK:Parameter für eine Beschreibung des tags IBK:Parameter):

Name	Vorgabe Einheit	Beschreibung	Wertebereich	Vorgabe	Verwendung
RelTol	---	Relative Toleranz für die Fehlerprüfung des Solvers.	0...0.1	1E-04	*optional*
AbsTol	---	Absolute Toleranz für die Fehlerprüfung des Solvers.	0...1	1E-10	*optional*
MaxTimeStep	h	Maximal zulässiger Zeitschritt für die Integration.	positiv double (> 0.0)	1	*optional*
MinTimeStep	s	Minimal akzeptierter Zeitschritt, bevor der Solver mit einem Fehler abbricht.	positive double (> 0.0)	1E-12	*optional*
InitialTimeStep	s	Initiale Zeitschrittgröße (oder konstante Schrittgröße für ExplicitEuler-Integrator).	positive double (> 0.0)	0.1	*optional*
NonlinSolverConvCoeff	---	Koeffizient, der die Konvergenzgrenze des Solvers nichtlinearer Gleichungen reduziert. Wird von Implicit Euler nicht unterstützt.	0...1	0.1	*optional*
IterativeSolverConvCoeff	---	Koeffizientenreduzierende Konvergenzgrenze des iterativen Gleichungssolvers.	0...1	0.05	*optional*
DiscMinDx	mm	Minimale Elementbreite für Wanddiskretisierung.	positiv double (> 0.0)	2	*optional*

Name	Vorgabe Einheit	Beschreibung	Wertebereich	Vorgabe	Verwendung
DiscStretchFactor	---	Stretch-Faktor für variable Wanddiskretisierungen: • **0** - keine Diskretisierung • **1** - äquidistant • > **1** - variabel siehe spatial discretization algorithm für Details.	positive integer (>= 0)	50	*optional*
(*)ViewfactorTileWidth	m	Maximale Abmessung einer Kachel für die Berechnung der Ansichtsfaktoren.	positive double (> 0.0)	50	*optional*
(*)SurfaceDiscretizationDensity	---	Anzahl der Oberflächendiskretisierungselemente einer Wand in jeder Richtung.	0...1	2	*optional*
(*)ControlTemperatureTolerance	K	Temperaturtoleranz für ideales Heizen oder Kühlen.	positiv double (> 0.0)	1E-05	*optional*
(*)KinsolRelTol	---	Relative Toleranz für Kinsol-Solver.	0...1	-	*optional*
(*)KinsolAbsTol	---	Absolute Toleranz für Kinsol-Löser.	0...1	-	*optional*

(*) - bisher noch nicht verwendet

Ganzzahlige Parameter (siehe Abschnitt IBK:IntPara für eine Beschreibung des tags IBK:IntPara):

Name	Beschreibung	Standard	Verwendung
PreILUWidth	Anzahl der Nicht-Nullen in ILU	---	*optional*
MaxKrylovDim	Max. Größe der Krylow-Dimension/max. Anzahl der linearen Iterationen (nur iterative LES)	50	*optional*
MaxNonlinIter	Max. Anzahl der nicht-linearen/Newton-Iterationen	3	*optional*
MaxOrder	Max. Methodenordnung	5	*optional*
DiscMaxElementsPerLayer	Max. Anzahl der Diskretisierungselemente pro Materialschicht	20	*optional*
(*)KinsolMaxNonlinIter	Max. Iterationen des Kinsol-Solvers	*auto*	*optional*

(*) - bisher noch nicht verwendet

Flags und Optionen (siehe Abschnitt IBK:Flag für eine Beschreibung des tags IBK:Flag):

Name	Beschreibung	Standard	Verwendung
(*)`DetectMaxTimeStep`	Zeitpläne prüfen, um Mindestabstände zwischen Schritten zu ermitteln und MaxTimeStep anzupassen.	*false*	*optional*
(*)`KinsolDisableLineSearch`	Deaktiviere Liniensuche für stationäre Zyklen.	*false*	*optional*
(*)`KinsolStrictNewton`	Strict Newton für stationäre Zyklen einschalten.	*false*	*optional*

(*) - bisher noch nicht verwendet

Die oben aufgeführten Optionen und Parameter hängen teilweise von den gewählten Zeitintegrationsalgorithmen, LES-Solvern und Vorkonditionierern ab, siehe Tabelle im Abschnitt Solver-Fähigkeiten unten.

Integrator

Der XML-tag `Integrator` enthält eine Zeichenkette zur Auswahl eines bestimmten Integrators (`CVODE` wird standardmäßig verwendet, wenn das tag fehlt).

Tabelle 22. verfügbare Integratoren

Name	Beschreibung
`CVODE`	Wählt den **CVODE**-Integrator aus der Sundials-Bibliothek: implizites Mehrschrittverfahren mit fehlertestbasierter Zeitschrittanpassung und modifiziertem Newton-Raphson für nichtlineare Gleichungssysteme
`ExplicitEuler`	Expliziter Euler-Integrator (nur zur Fehlersuche, der Parameter `InitialTimeStep` bestimmt die feste Schrittweite)
`ImplicitEuler`	Impliziter Euler-Integrator, Einzelschrittlöser mit fehlertestbasierter Zeitschrittanpassung und modifiziertem Newton-Raphson für nichtlineare Gleichungssysteme (nur zur Fehlersuche und für spezielle Tests)

Siehe Solver-Fähigkeiten für gültige Kombinationen.

Linear equation system (LES) solver

Der XML-tag `LesSolver` enthält eine Zeichenkette zur Auswahl eines bestimmten Solvers für die linearen Gleichungssysteme (`KLU` wird standardmäßig verwendet, wenn der tag fehlt).

Tabelle 23. verfügbare LES-Solver

Name	Beschreibung
`Dense`	Direkter dense Solver (nur zur Fehlersuche)
`KLU`	Direkter Sparse Solver
`GMRES`	Verallgemeinerte Minimale Residualmethode (iterativer Solver)
`BiCGStab`	Bikonjugierte stabilisierte Gradientenmethode (iterativer Solver)

Siehe Solver-Fähigkeiten für gültige Kombinationen.

Präkonditionierer

Der XML-tag `Preconditioner` enthält eine Zeichenkette zur Auswahl eines bestimmten Preconditioners, der für iterative LES-Solver verwendet werden soll (`ILU` wird standardmäßig verwendet, wenn das tag fehlt).

Tabelle 24. verfügbare Preconditioners

Name	Beschreibung
`ILU`	Unvollständige LU-Faktorisierung (wenn `PreILUWidth` angegeben ist, wird ILU-T verwendet)

Derzeit sind zwei Varianten des ILU-Preconditioners implementiert. Eine ohne Schwellenwert, bei der die Faktorisierung nur im ursprünglichen Jacobi-Matrixmuster gespeichert wird. Wenn der Benutzer `PreILUWidth` angegeben hat, berechnet die Routine die Faktorisierung und behält in jeder Zeile die höchsten n-Werte (wobei n durch `PreILUWidth` definiert ist). Diese Methode ist bekannt als *ILU mit Threashold* (ILU-T).

Eine ILU-T-Methode ist nur für `PreILUWidth` > 3 wirksam. Die minimale Anzahl von Nicht-Nullen in jeder Matrixzeile ist 3, da die Finite-Volumen-Diskretisierung der Wandkonstruktionen bereits ein 3-Diagonal-Muster erzeugt.

Solver-Fähigkeiten

Nicht alle Integratoren und LES-Solver unterstützen alle oben genannten Optionen. Auch können nicht alle LES-Solver mit allen Integratoren kombiniert werden. Die folgende Tabelle gibt einen Überblick über die unterstützten Kombinationen und Optionen.

Tabelle 25. Fähigkeiten und unterstützte Flags/Parameter für die angebotenen Integratoren

Integrator	LES-Solver	Unterstützte Integratorparameter/Flags
CVODE	Dense, KLU, GMRES, BiCGStab	RelTol, AbsTol, MaxTimeStep, MinTimeStep, InitialTimeStep, MaxOrder, NonlinSolverConvCoeff, MaxNonlinIter
ImplicitEuler	Dense	RelTol, AbsTol, MaxTimeStep, InitialTimeStep, NonlinSolverConvCoeff, MaxNonlinIter
ExplicitEuler	---	InitialTimeStep

Tabelle 26. Fähigkeiten und unterstützte Flags/Parameter für die angebotenen LES-Solver

LES-Solver	Preconditioners	Unterstützte Integratorparameter/Flags
DENSE	---	---
KLU	---	---
GMRES	ILU	PreILUWidth, MaxKrylovDim, IterativeSolverConvCoeff
BiCGStab	ILU	PreILUWidth, MaxKrylovDim, IterativeSolverConvCoeff

2.16. Modellparametrisierung

In diesem Abschnitt werden die verschiedenen Modellparametrisierungsblöcke beschrieben. Modelle werden

verwendet, um gleichartige Funktionalität für mehrere Objekte (meist Zonen) zu definieren. Die in den Modellblöcken abgelegten Parameter gelten dann für alle (via Objektlisten) ausgewählten Objekte. Allerdings können objektspezifische Eigenschaften (bspw. Nutzfläche bei Zonen) in die Modell einfließen.

2.16.1. Natürliches Lüftungsmodell

Das Modell für natürliche Lüftung definiert den Luftaustausch mit der Außenluft und beinhaltet Nutzerlüftung wie ungewollte Lüftung durch Leckagen (Fugenlüftung/Infiltration). Die natürliche Lüftung wird für bestimmte Zonen aktiviert, indem ein Modellparametersatz `NaturalVentilationModel` definiert wird. Über die Objektliste werden die Zonen ausgewählt.

```
<!-- Lüftungsmodell mit durchgängig konstanter Infiltration/Grundluftwechsel -->
<NaturalVentilationModel id="501" displayName="Zone vent" modelType="Constant">
  <ZoneObjectList>Office zones</ZoneObjectList>
  <IBK:Parameter name="VentilationRate" unit="1/h">0.5</IBK:Parameter>
</NaturalVentilationModel>
<!-- Lüftungsmodell mit zeitabhängig geregelter Luftwechselrate -->
<NaturalVentilationModel id="502" displayName="Zone vent" modelType="Scheduled">
  <ZoneObjectList>Other zones</ZoneObjectList>
</NaturalVentilationModel>
<!-- Lüftungsmodell mit zeitabhängig geregelter, erhöhter Luftwechselrate und Grundluftwechsel -->
<NaturalVentilationModel id="503" displayName="Zone vent" modelType="ScheduledWithBaseACR">
  <IBK:Parameter name="VentilationRate" unit="1/h">0.5</IBK:Parameter>
  <!-- Erhöhte Lüftungsrate verwenden, wenn Raumtemperature 23°C überschreitet -->
  <IBK:Parameter name="MinimumRoomAirTemperatureACRLimit" unit="C">23</IBK:Parameter>
  <!-- Keine Begrenzung der erhöhten Lüftung bei hohen Raumtemperaturen  -->
  <IBK:Parameter name="MaximumRoomAirTemperatureACRLimit" unit="C">100</IBK:Parameter>
  <!-- Parameter muss gegeben sein, um Regelung auszuschalten, sehr hohe Grenzwerte verwenden -->
  <IBK:Parameter name="MaximumEnviromentAirTemperatureACRLimit" unit="C">100</IBK:Parameter>
  <IBK:Parameter name="MinimumEnviromentAirTemperatureACRLimit" unit="C">-100</IBK:Parameter>
  <IBK:Parameter name="DeltaTemperatureACRLimit" unit="K">0</IBK:Parameter>
  <IBK:Parameter name="WindSpeedACRLimit" unit="m/s">10</IBK:Parameter>
  <ZoneObjectList>Special zones</ZoneObjectList>
</NaturalVentilationModel>

<!-- Lüftungsmodell mit zeitabhängig geregelter, erhöhter Luftwechselrate und Grundluftwechsel,
     Regelparameter werden über Schedules geliefert.
-->
<NaturalVentilationModel id="504" displayName="Zone vent" modelType="ScheduledWithBaseACR">
  <IBK:Parameter name="VentilationRate" unit="1/h">0.5</IBK:Parameter>
  <ZoneObjectList>Special zones</ZoneObjectList>
</NaturalVentilationModel>

<!-- Lüftungsmodell mit realistischer Berechnung der Luftwechselrate basierend auf
     Windgeschwindigkeit und Temperaturgradienten -->
<NaturalVentilationModel id="505" displayName="Zone vent" modelType="Realistic">
  <IBK:Parameter name="VentilationRate" unit="1/h">0.5</IBK:Parameter>
  <IBK:Parameter name="ConstantCoefficient" unit="-">0.6</IBK:Parameter>
  <IBK:Parameter name="TemperaturCoefficient" unit="1/K">0.04</IBK:Parameter>
  <IBK:Parameter name="WindCoefficient" unit="s/m">0.04</IBK:Parameter>
  <IBK:Parameter name="SquaredWindCoefficient" unit="s2/m2">0.0</IBK:Parameter>
  <ZoneObjectList>Special other zones</ZoneObjectList>
</NaturalVentilationModel>
```

Es darf nur ein Lüftungsmodellblock für eine Zone definiert werden. Es dürfen also nicht zwei Lüftungsmodellblöcke mit Objektlisten definiert werden, die beide die gleiche(n) Zone(n) enthalten.

Das NaturalVentilationModel muss mit den folgenden XML-Attributen definiert werden:

Tabelle 27. Attribute

Attribut	Beschreibung	Format	Verwendung
id	Kennung des natürlichen Lüftungsmodells	> 0	*erforderlich*
modelType	Setzt den Typ des Lüftungsmodells • Constant - Konstante Luftwechselrate • Scheduled - Luftwechselrate ändert sich nach einem definierten Zeitplan. Zusätzlich kann durch Steuerungsmodelle der Luftwechsel auf null gesetzt werden. • ScheduledWithBaseACR - Es wird ein konstanter Luftwechsel und ein zeitabhängiger Luftwechsel - Scheduled with base air change rate - integriert. Dabei kann der zeitabhängige Luftwechsel zusätzlich durch Steuerungsmodelle auf null gesetzt werden. • Realistic- TODO	key	*erforderlich*

Gleitkommazahlen-Parameter (siehe Abschnitt IBK:Parameter für eine Beschreibung des tags IBK:Parameter):

Name	Vorgabeeinheit	Beschreibung	Wertebereich	Verwendung
VentilationRate	1/h	Konstante Luftwechselrate	>= 0.0	*erforderlich für Modelltyp* Constant oder ScheduledWithBaseACR
*ConstantCoefficient	-	Modifikationsfaktor C1 für die Luftwechselrate (Standardwert 1)	>= 0.0	*optional*
*TemperatureCoefficient	1/K	Modifikationsfaktor C2 für die Luftwechselrate (Standardwert 0)	>= 0.0	*optional*
*WindCoefficient	s/m	Modifikationsfaktor C3 für die Luftwechselrate (Standardwert 0)	>= 0.0	*optional*
*SquaredWindCoefficient	s2/m2	Modifikationsfaktor C4 für die Luftwechselrate (Standardwert 0)	>= 0.0	*optional*

(*) noch nicht implementiert

Beim Modelltyp Constant wird durchweg eine konstante Luftwechselrate (Parameter VentilationRate) verwendet.

Beim Modelltyp Scheduled wird die Luftwechselrate aus einem Zeitplan (Parameter VentilationRateSchedule, siehe

Zeitpläne) entnommen. Weitere Parameter sind nicht notwendig.

Nachfolgend finden Sie ein Beispiel für einen Zeitplan, der den Parameter `VentilationRateSchedule` für ein solches Modell der geplanten natürlichen Lüftung bereitstellt:

Beispiel 47. Zeitplan, der den Parameter VentilationRateSchedule bereitstellt

```
<ScheduleGroup objectList="All zones">
  <!-- jeden Tag zwischen 8-10 -->
  <Schedule type="AllDays">
    <DailyCycles>
      <DailyCycle interpolation="Constant">
        <TimePoints>0 6 10</TimePoints>
        <Values>VentilationRateSchedule [1/h]:0 0.4 0</Values>
      </DailyCycle>
    </DailyCycles>
  </Schedule>
  <!-- Dienstag keine Lüftung -->
  <Schedule type="Tuesday">
    <DailyCycles>
      <DailyCycle interpolation="Constant">
        <TimePoints>0</TimePoints>
        <Values>VentilationRateSchedule [1/h]:0</Values>
      </DailyCycle>
    </DailyCycles>
  </Schedule>
  <!-- Wochenende nur am Nachmittag -->
  <Schedule type="WeekEnd">
    <DailyCycles>
      <DailyCycle interpolation="Constant">
        <TimePoints>0 14 16</TimePoints>
        <Values>VentilationRateSchedule [1/h]:0 0.1 0</Values>
      </DailyCycle>
    </DailyCycles>
  </Schedule>
</ScheduleGroup>
```

Beim Modelltyp `ScheduledWithBaseACR` wird ein konstanter Grundluftwechsel (Parameter `VentilationRate`) verwendet und unter bestimmten Bedingungen wird ein zusätzlicher Luftwechsel `VentilationRateSchedule` verwendet.

Die Luftwechselrate wird berechnet:

```
n = n_Grundluftwechsel                          // wenn Bedingungen nicht erfüllt
n = n_Grundluftwechsel + n_erhöhterLuftwechsel  // wenn Bedingungen erfüllt
```

Regelbedingungen

Folgende Bedingungen müssen *alle* erfüllt sein, damit der erhöhte Luftwechsel addiert wird.

- Raumluftzustand: `MinimumRoomAirTemperatureACRLimit` < T_Raum < `MaximumRoomAirTemperatureACRLimit`
- Außentemperatur: `MinimumEnviromentAirTemperatureACRLimit`< T_Außenluft < `MaximumEnviromentAirTemperatureACRLimit`
- Temperaturdifferenz: T_Raum - T_Außenluft < `DeltaTemperatureACRLimit`, dabei kann `DeltaTemperatureACRLimit` auch negativ sein. Damit können auch Heizeffekte (Raum kalt, Außenluft warm) berücksichtigt werden.

- Windgeschwindigkeit: akt. Wingeschwindigkeit < `WindSpeedACRLimit`

Damit das Modell mit der Ausprägung `ScheduledWithBaseACR` verwendbar ist, müssen beide Parameter *VentilationRate* und *VentilationRateSchedule* spezifiziert werden.

Gleitkommazahlen-Parameter (siehe Abschnitt IBK:Parameter für eine Beschreibung des tags `IBK:Parameter`):

Name	Vorgabeeinheit	Beschreibung	Wertebereich	Verwendung
`MaximumRoomAirTemperatureACRLimit`	C	Maximale Raumlufttemperatur für aktive Lüftung.	-100 ... 100	*xxx*
`MinimumRoomAirTemperatureACRLimit`	C	Minimale Raumlufttemperatur für aktive Lüftung.	-100 ... 100	*xxx*
`MaximumEnviromentAirTemperatureACRLimit`	C	Maximale Außenlufttemperatur für aktive Lüftung.	-100 ... 100	*xxx*
`MinimumEnviromentAirTemperatureACRLimit`	C	Minimale Außenlufttemperatur für aktive Lüftung.	-100 ... 100	*xxx*
`DeltaTemperatureACRLimit`	K	Minimale Differenz von Raumlufttemperatur minus Außenlufttemperatur (Tzone - Tout) für aktive Lüftung.	-100 ... 100	*xxx*
`WindSpeedACRLimit`	m/s	Maximale Windgeschwindigkeit für aktive Lüftung.	0 ... 40	*xxx*

Die dazugehörigen zeitabhängigen Schedule-Parameter sind gleichartig benannt, jedoch jeweils mit dem Suffix `Schedule`.

Eine allgemeine Abhängigkeit von Wind- und/oder Temperaturpotentialen ist möglich, wenn man für einen odere mehrere Koeffizienten Abweichungen von dem/den Standardwert(en) setzt. Die allgemeine Formel zur modifizierten Luftwechselrate lautet:

$n = n * (C1 + C2 * (Tzone - Tout) + C3 * windSpeed + C4 * windSpeed^2)$

Die Modifikation der Luftwechselrate wird immer auf beide Luftwechselraten (konstant und zeitgesteuert) angewendet.

TODO : Spezifizieren

Ausgabegrößen

Die berechneten Lüftungswärmeverluste/-gewinne können als zonenspezifische Größen über die Ergebnisgröße `Zone.VentilationHeatLoad` in [W] abgerufen werden.

2.16.2. Steuerungsmodell für Verschattung

Ein Verschattungregelungsmodell ist eine spezielle Art von Regelungsmodell, das einen Signalwert zwischen 0 (keine Verschattung) und 1 (volle Verschattung) zurückgibt. Das tatsächliche Ausmaß der Verschattung bzw. die Reduzierung der solaren Gewinne wird durch den Verschattungs-Parameterblock (`Shading`, siehe Fensterverschattung) bestimmt. Somit kann das gleiche Regelmodell für verschiedene Verschattungseinrichtungen

verwendet werden.

Beispiel 48. Parameterdefinition für Verschattungsregelungsmodell

```
<Models>
    <ShadingControlModels>
      <!-- ShadingControlModel liefert einen Wert zwischen 0 und 1
        0 = keine Reduktion (Verschattung offen)
        1 = volle Reduktion (Verschattung geschlossen)
      -->
      <ShadingControlModel id="2000" displayName="Global horizontal sensor controller" sensorID="50000">
        <IBK:Parameter name="MaxIntensity" unit="W/m2">300</IBK:Parameter>
        <IBK:Parameter name="MinIntensity" unit="W/m2">150</IBK:Parameter>
      </ShadingControlModel>
    </ShadingControlModels>
</Models>
```

Das Verschattungskontrollmodell verlangt 2 Parameter `MaxIntensity` und `MinIntensity` und implementiert eine digitale Regelung mit Hysterese. Zunächst muss die Globalstrahlungsintensität auf den Sensor den oberen Grenzwert überschreiten `MaxIntensity`, wonach die Verschattung geschlossen wird (Kontrollmodell liefert 1). Danach muss die Strahlungsintensität zunächst unter die untere Grenze sinken (`MinIntensity`), bevor die Verschattung wieder geöffnet wird (Kontrollmodell liefert 0).

Für die Auswertung wird eine Horizontalstrahlung benötigt. Dafür muss eine Oberfläche ausgewählt werden und als `sensorID` angegeben werden. Möglich sind hier 3 Optionen:

- allgemeiner Sensor auf einer Fläche (siehe Zusätzliche Strahlungssensoren)
- ID eines Fensters (eigentlich ID des *embedded object*, welches das Fenster enthält); hier wird die Globalstrahlung durch das Fenster als Eingangsgröße verwendet, einschließlich eventueller externer Verschattung bzw. Eigenverschattung
- ID einer opaquen Fläche; hier wird die Globalstrahlung auf eine opaque Fläche als Eingangsgröße verwendet, einschließlich eventueller externer Verschattung bzw. Eigenverschattung

Damit diese IDs eindeutig auflösbar sind, müssen Sensoren, Fenster und Konstruktionen global eindeutige IDs tragen.

Ausgabegrößen

Das Verschattungssteuerungsmodell liefert als Ergebnisgrößen:

- `ShadingControlValue` - Steuerungssignal für Verschattung: 0 - komplett offen, 1 - komplett geschlossen, Zwischenwerte sind möglich
- `SolarIntensityOnShadingSensor` - Solarstrahlungsintensität auf ausgewählten Sensor, der für die Regelung verwendet wird

2.16.3. Modell für interne Lasten

Das interne Lastenmodell wird verwendet, um die Wärmelasten von Geräten, Personen und Beleuchtung für Zonen zu definieren. Interne Lasten werden genauso definiert wie natürliche Lüftungsmodelle. Der Objektlisten-tag `ZoneObjectList` identifiziert die Zonen, in denen interne Lasten berücksichtigt werden sollen. Es dürfen nicht zwei interne Lastmodelle existieren, die sich auf dieselben Zonen beziehen (nur eine interne Last pro Zone).

Beispiel 49. Definitionsblock für interne Lasten

```
<InternalLoadsModel id="200" modelType="Scheduled">
  <ZoneObjectList>Office zones</ZoneObjectList>
  <IBK:Parameter name="RadiantFraction" unit="---">0.5</IBK:Parameter>
</InternalLoadsModel>
```

Das `InternalLoadsModel` muss mit den folgenden XML-Attributen definiert werden:

Tabelle 28. Attribute

Attribut	Beschreibung	Format	Verwendung
`id`	Kennung des Modells	> 0	*erforderlich*
`modelType`	Gibt an, wie die internen Lasten angesetzt werden sollen • `Constant` - Konstante Geräte-, Personen- und Beleuchtungsenergielasten • `Scheduled` - Lasten werden über Zeitplanparameter bereitgestellt.	key	*erforderlich*

Fließkommaparameter (siehe Abschnitt IBK:Parameter für eine Beschreibung des Tags `IBK:Parameter`):

Name	Vorgabeeinheit	Beschreibung	Wertebereich	Verwendung
`EquipmentHeatLoadPerArea`	W/m2	Komplette Gerätebelastung pro Zonennutzfläche	>= 0.0	*erforderlich für Konstantes Modell*
`PersonHeatLoadPerArea`	W/m2	Komplette Personenwärmelast pro Zonennutzfläche	>= 0.0	*erforderlich für Konstantes Modell*
`LightingHeatLoadPerArea`	W/m2	Komplette Wärmelast aus Beleuchtung pro Zonennutzfläche	>= 0.0	*erforderlich für Konstantes Modell*
`EquipmentRadiationFraction`	---	Prozentualer Anteil der Wärme der Geräte, der durch Strahlung emittiert wird	>= 0.0	*erforderlich*
`PersonRadiationFraction`	---	Prozentualer Anteil der Wärme der Personen, der durch Strahlung emittiert wird	>= 0.0	*erforderlich*

Name	Vorgabeeinheit	Beschreibung	Wertebereich	Verwendung
LightingRadiationFraction	---	Prozentualer Anteil der Wärme der Beleuchtung, der durch Strahlung emittiert wird	>= 0,0	*erforderlich*

Die Zonennutzfläche ist nicht zwingend die Grundfläche einer Zone sondern wird aus dem Parameter *Area* der Zonendefinition gewählt. Dadurch ist es möglich, z.B. im Dachgeschoss mit Schrägen die tatsächlich nutzbare Fläche zu definieren und zu verwenden. Deshalb wird der *Area* Parameter in allen Zonen benötigt, für die ein `InternalLoadsModel` angewendet werden soll.

Der Modelltyp `Constant` übernimmt die internen Lasten aus den Parametern (siehe oben). Wenn der Modelltyp `Scheduled` verwendet wird, werden die tatsächlichen Lasten aus dem Zeitplan entnommen.

Die folgenden Zeitplanparameter sind erforderlich:

- `EquipmentHeatLoadPerAreaSchedule [W/m2]`
- `PersonHeatLoadPerAreaSchedule [W/m2]`
- `LightingHeatLoadPerAreaSchedule [W/m2]`

Die zonenspezifischen Ausgangsgrößen heißen:

- `ConvectiveEquipmentHeatLoad [W]`
- `ConvectivePersonHeatLoad [W]`
- `ConvectiveLightingHeatLoad [W]`
- `RadiantEquipmentHeatLoad [W]`
- `RadiantPersonHeatLoad [W]`
- `RadiantLightingHeatLoad [W]`

Dies sind vektoriell dargestellte Größen, die in Ausgangsdefinitionen referenziert werden müssen, z. B. mit: `ConvectiveEquipmentHeatLoad[id=3]` für die konvektive Gerätelast in Zone #3.

Die Parameter `xxxRadiationFraction` geben an, welcher Prozentsatz der berechneten internen Lasten als Strahlungsfluss flächengewichtet auf opake Oberflächen, die die Zone umschließen, aufgebracht werden soll.

2.16.4. Modell für Thermostate

Das Thermostatmodell beschreibt, auf welche Raumsollwerte konditioniert werden soll. Angegeben werden können Heiz- und/oder Kühlsolltemperaturen für die Raumluft oder operative Raumluft. Der Objektlisten-tag `ZoneObjectList` identifiziert die Zonen, in denen Thermostate berücksichtigt werden sollen. Es darf nur ein Modell pro Zone existieren.

Beispiel 50. Definitionsblock für Thermostate

```
<ZoneControlThermostat id="200" modelType="Constant">
  <ZoneObjectList>Office zones</ZoneObjectList>
  <IBK:Parameter name="HeatingSetpoint" unit="C">20</IBK:Parameter>
  <IBK:Parameter name="CoolingSetpoint" unit="C">21</IBK:Parameter>
  <TemperatureType>AirTemperature</TemperatureType>
</ZoneControlThermostat>
```

Das `ZoneControlThermostat` muss mit den folgenden XML-Attributen definiert werden:

Tabelle 29. Attribute

Attribut	Beschreibung	Format	Verwendung
`id`	Kennung des Modells	> 0	*erforderlich*
`modelType`	Gibt an, wie die Thermostat-Parameter angesetzt werden sollen • `Constant` - Konstante Sollwerte • `Scheduled` - Sollwerte werden über Zeitplanparameter bereitgestellt.	key	*erforderlich*

Fließkommaparameter (siehe Abschnitt IBK:Parameter für eine Beschreibung des Tags `IBK:Parameter`):

Name	Vorgabeeinheit	Beschreibung	Wertebereich	Verwendung
`HeatingSetpoint`	C	konstanter Heizsollwert	< `CoolingSetpoint`	*erforderlich für Modelltyp* `Constant`
`CoolingSetpoint`	C	konstanter Kühlsollwert	> `HeatingSetpoint`	*erforderlich für Modelltyp* `Constant`

TemperatureType

Der XML-tag `TemperatureType` enthält eine Zeichenkette zur Auswahl eines bestimmten Typs (`AirTemperature` wird standardmäßig verwendet, wenn das tag fehlt).

Tabelle 30. verfügbare TemperatureTypes

Name	Beschreibung
`AirTemperature`	Als Referenztemperatur wird die Raumlufttempatur verwendet.

Name	Beschreibung
OperativeTemperature	Als Referenztemperatur wird die operative Raumlufttempatur verwendet. Diese setzt sich aus der mittleren Oberflächentemperatur aller Innenoberflächen und aus der Raumlufttemperatur zusammen. Die Anteile betragen jeweils 50%.

Ein Thermostat hält nur die Sollwerte für die Zone. Eine Konditionierung der Zone erfolgt erst wenn zusätzlich eine Heizungs- und/oder Kühlmodell für die Zone integriert ist. Auch bei den Zeitplänen ist immer darauf zu achten, dass der Heizsollwert < Kühlsollwert ist.

Der Modelltyp `Constant` übernimmt die Sollwerte aus den Parametern (siehe oben). Wenn der Modelltyp `Scheduled` verwendet wird, werden die tatsächlichen Sollwerte aus dem Zeitplan entnommen.

Die folgenden Zeitplanparameter sind erforderlich:

- `HeatingSetpointSchedule [C]`
- `CoolingSetpointSchedule [C]`

Die zonenspezifischen Ausgangsgrößen heißen:

- `HeatingSetpoint [C]`
- `CoolingSetpoint [C]`

Dies sind vektoriell dargestellte Größen, die in Ausgangsdefinitionen referenziert werden müssen, z. B. mit: `HeatingSetpoint[id=3]` für den Heizsollwert in Zone #3.

2.16.5. Modell für ideale thermische Konditionierung

Das Modell beschreibt ein ideles thermisches Konditionierungsmodell für eine ideale Raumluftkonditionierung. Der Objektlisten-tag `ZoneObjectList` identifiziert die Zonen, in denen das Modell berücksichtigt werden sollen. Es darf nur ein Modell pro Zone existieren.

Beispiel 51. Definitionsblock für Ideale thermische Konditionierung

```
<ZoneIdealHeatingCooling id="200">
  <ZoneObjectList>Office zones</ZoneObjectList>
  <IBK:Parameter name="MaxHeatingPowerPerArea" unit="W/m2">50</IBK:Parameter>
  <IBK:Parameter name="MaxCoolingPowerPerArea" unit="W/m2">20</IBK:Parameter>
</ZoneIdealHeatingCooling>
```

Das `ZoneIdealHeatingCooling` muss mit den folgenden XML-Attributen definiert werden:

Tabelle 31. Attribute

Attribut	Beschreibung	Format	Verwendung
id	Kennung des Modells	> 0	*erforderlich*

Fließkommaparameter (siehe Abschnitt IBK:Parameter für eine Beschreibung des tags `IBK:Parameter`):

Name	Vorgabeeinheit	Beschreibung	Wertebereich	Verwendung
`MaxHeatingPowerPerArea`	W/m2	maximale flächenbezogene Heizleistung	>= 0.0	*erforderlich*
`MaxCoolingPowerPerArea`	W/m2	maximale flächenbezogene Kühlleistung	>= 0.0	*erforderlich*

Damit das Modell auf die jeweilige Zone angewendet wird, ist zwingend das Modell für Thermostate nötig, welches für die gleichen Zone parametriert sein muss. Dieses liefert als Eingangsgrößen für das `ZoneIdealHeatingCooling` die Größen `HeatingSetpoint` und `CoolingSetpoint`.

Regelungsimplementierung

Standardmäßig verwendet das ideale Konditionierungsmodell einen P-Regler, nach dem Schema:

```
Heizlast = MaxHeatingPowerPerArea * clip( (T_soll - T_raum) / DeltaT )

// alternativ mit Mittelpunktsverschiebung
Heizlast = MaxHeatingPowerPerArea * clip( (T_soll - T_raum) / DeltaT + 0.5))
```

`clip` heißt: zwischen 0 und 1 begrenzen.

`T_raum` ist entweder die Lufttemperatur oder die operative Temperatur. Dafür ist die Information im ThermostatModell notwendig.

Ausgaben

Die zonenspezifischen Ausgangsgrößen heißen:

- `IdealHeatingLoad [W]`
- `IdealCoolingLoad [W]`

Dies sind vektoriell dargestellte Größen, die in Ausgangsdefinitionen referenziert werden müssen, z. B. mit: `IdealHeatingLoad[id=3]` für die Heizlast in Zone #3.

2.17. HydraulicNetworks

Dieser Abschnitt beschreibt die Parametrisierung von thermo-hydraulischen Netzwerken.

In einer Projektdatei kann es mehrere Netzwerke geben. Diese sind im XML-tag `HydraulicNetworks` abgelegt.

Beispiel 52. Hydraulische Netzwerke

```
<HydraulicNetworks>
  <HydraulicNetwork id="1" displayName="xxx">
        ... <!-- network parameters -->
  </HydraulicNetwork>

  <HydraulicNetwork id="2" displayName="yyy">
        ... <!-- network parameters -->
  </HydraulicNetwork>

    ... <!-- other networks -->

</HydraulicNetworks>
```

Die einzelnen Netzwerke sind hydraulisch unabhängig, d.h. es gibt keinen Fluidaustausch zwischen Netzwerken. Die Netzwerke können aber thermisch verknüpft sein, z.B. kann ein Quartierswärmenetz mit einem Gebäudeanlagennetz Wärme austauschen. Dies bedingt auch die *eindeutige ID* der in *allen* Netzwerken definierten hydraulischen Netzwerkkomponenten.

2.17.1. Definition eines hydraulischen Netzwerks

Ein einzelnes Netzwerk wird im XML-tag `HydraulicNetwork` definiert, wie im folgenden Beispiel:

Beispiel 53. Typische Definition eines einzelnen Netzwerks

```
<HydraulicNetwork id="1" displayName="Simple network" modelType="ThermalHydraulicNetwork"
        referenceElementId="201">
  <HydraulicFluid id="1" displayName="Water">
        ... <!-- fluid parameters -->
    </HydraulicFluid>

  <PipeProperties>
        ... <!-- database with pipe definitions -->
  </PipeProperties>

  <Components>
        ... <!-- database with component definitions -->
  </Components>

  <Elements>
        ... <!-- network flow elements with topology definition -->
  </Elements>

</HydraulicNetwork>
```

Der XML-tag `HydraulicNetwork` hat folgende Attribute:

Tabelle 32. Attributes

Attribut	Beschreibung	Format	Verwendung
`id`	Eindeutige Identifizierungsnummer	> 0	*benötigt*

Attribut	Beschreibung	Format	Verwendung
displayName	Anzeigename/Beschreibung des Netzwerks	string	*optional*
modelType	Legt das Berechnungsmodell fest	string	*optional* (defaults to ThermalHydraulicNetwork)
referenceElementId	ID des Referenzelements	> 0	*benötigt*

Das optionale Attribut modelType beschreibt, welches Berechnungsmodell zu verwenden ist (Tabelle 33).

Tabelle 33. Werte für Attribut modelType

Attribut	Beschreibung
HydraulicNetwork	Nur die hydraulische Berechnung (Masseströme und Drücke) wird durchgeführt.
ThermalHydraulicNetwork	Netzwerk wird mit eingeschalteten Energiebilanzen berechnet. Jedes Strömungselement entspricht mindestens einer Energiebilanzgleichung.

Die Wahl des Netzwerkberechnungsmodells hat Auswirkungen auf die benötigte Parametrisierung der beteiligten Komponenten.

Parameter

Je nach Modelltyp können mehrere globale Netzwerkparameter definiert werden (siehe Abschnitt IBK:Parameter für eine Beschreibung des IBK:Parameter tags):

Name	Einheit	Beschreibung	Wertebereich	Verwendung
DefaultFluidTemperature	C	einheitliche, konstante Fluidtemperatur zur Verwendung mit einem rein hydraulischen Modell	> 0.0	*benötigt für Modelltyp* HydraulicNetwork
InitialFluidTemperature	C	Anfangsfluidtemperatur	> 0.0	*benötigt für Modelltyp* ThermalHydraulicNetwork
ReferencePressure	Pa	Druck hinter dem Referenzelement	> 0.0	*optional* (Standardwert ist 0 Pa)

Innerhalb des Netzwerk-Definitionsblocks gibt es vier Kind-Elemente:

- HydraulicFluid - definiert Medieneigenschaften
- PipeProperties - Rohreigenschaften
- Components - Komponentendefinitionen
- Elements - Netzwerkelemente und Netzwerktopologie

2.17.2. Medieneigenschaften

Der XML-tag `HydraulicFluid` enthält die Definition des im Netzwerk strömenden Mediums.

In jedem Netzwerk gibt es nur ein einziges Fluid, und deshalb ist auch nur eine einzige Instanz des `HydraulicFluid` Tags erlaubt.

Beispiel 54. Definition eines Netzwerkfluids

```
<HydraulicFluid id="1" displayName="Water">
  <IBK:Parameter name="Density" unit="kg/m3">998</IBK:Parameter>
  <IBK:Parameter name="HeatCapacity" unit="J/kgK">4180</IBK:Parameter>
  <IBK:Parameter name="Conductivity" unit="W/mK">0.6</IBK:Parameter>
  <LinearSplineParameter name="KinematicViscosity" interpolationMethod="linear">
    <X unit="C">0 10 20 30 40 50 60 70 80 </X>
    <Y unit="m2/s">1.793e-06 1.307e-06 1.004e-06 8.01e-07 6.58e-07 5.54e-07 4.75e-07 4.13e-07 3.65e-07 </Y>
  </LinearSplineParameter>
</HydraulicFluid>
```

Der XML-tag `HydraulicFluid` hat folgende Attribute:

Tabelle 34. Attributes

Attribut	Beschreibung	Format	Verwendung
`id`	Eindeutige Identifikationsnummer	> 0	*benötigt*
`displayName`	Beschreibung des Fluids	string	*optional*

Da es nur eine Fluid-Definition per Netzwerk geben darf, ist das *id* Attribut eigentlich überflüssig. Es wird lediglich zur Aktualisierung der Programmdatenbank beim Import eines Projekts benötigt.

Parameter des Netzwerkfluids (siehe Abschnitt IBK:Parameter für eine Beschreibung des `IBK:Parameter` tags):

Name	Einheit	Beschreibung	Wertebereich	Verwendung
`Density`	kg/m3	Dichte bei Referenztemperatur	> 0.0	*benötigt*
`HeatCapacity`	J/kgK	Spezifische Wärmekapazität	> 0.0	*benötigt*
`Conductivity`	W/mK	Wärmeleitfähigkeit bei Referenztemperatur	>= 0.0	*benötigt*

Die obigen Eigenschaften, insbesondere die Dichte, werden zur Vereinfachung als temperaturunabhängig konstant angenommen. Für die meisten Anwendungsfälle der thermo-hydraulischen Simulation im Gebäude-/Quartierskontext wird die thermische Ausdehnung des Fluids nicht benötigt. Und die Auslegung des Ausdehngefäßes erfolgt nicht mit der Simulation.

Desweiteren gibt es noch temperaturabhängige Parameter, welche in linear interpolierten Datentabellen abgelegt werden (siehe Abschnitt LinearSplineParameter für eine Beschreibung des `LinearSplineParameter` Elements):

Name	Einheit	Beschreibung	Wertebereich	Verwendung
KinematicViscosity	m2/s	Kinematische Viscosität	> 0.0	*benötigt*

2.17.3. Rohreigenschaften

Die Rohreigenschaften legen die physikalische/geometrischen Eigenschaften eines Rohrtyps fest. Diese werden in XML-tags we HydraulicNetworkPipeProperties im Katalog PipeProperties mit eindeutigen IDs aufgelistet.

Beispiel 55. Definition von Rohreigenschaften

```
<PipeProperties>
  <HydraulicNetworkPipeProperties id="1">
    <IBK:Parameter name="PipeRoughness" unit="m">0.007</IBK:Parameter>
    <IBK:Parameter name="PipeInnerDiameter" unit="mm">25.6</IBK:Parameter>
    <IBK:Parameter name="PipeOuterDiameter" unit="mm">32</IBK:Parameter>
    <IBK:Parameter name="UValuePipeWall" unit="W/mK">5</IBK:Parameter>
  </HydraulicNetworkPipeProperties>

  ...
</PipeProperties>
```

Rohreigenschaften werden über das Attribut pipePropertyId eines Netzwerkelements (siehe Section 2.17.5) referenziert.

Tabelle 35. Attribute

Attribut	Beschreibung	Format	Verwendung
id	Eindeutige Identifikationsnummer des Rohrdatensatzes	> 0	*benötigt*

Parameter der Rohreigenschaftem (siehe Abschnitt IBK:Parameter für eine Beschreibung des IBK:Parameter Tags):

Name	Einheit	Beschreibung	Wertebereich	Verwendung
PipeRoughness	mm	Rauhheit der inneren Rohroberfläche	> 0.0	*benötigt*
PipeInnerDiameter	mm	Innendurchmesser des Rohres	> 0.0	*benötigt*
PipeOuterDiameter	mm	Außendurchmesser des Rohres	> 0.0	*benötigt*
UValuePipeWall	W/mK	Längenbezogener äquivalenter U-Wert der Rohrwand (einschließlich Dämmung, wenn vorhanen)	> 0.0	*benötigt (für Rohre mit Wärmeleitung nach Außen)*

Der Außendurchmesser muss größer als der Innendurchmesser sein.

Der längenbezogene äquivalente U-Wert der Rohrwand (einschließlich möglicher Dämmung) ist in der Berechnung

so definiert, dass eine Multiplikation mit der Temperaturdifferenz zwischen Fluidtemperatur und Außentemperatur zum Wärmeström pro m Rohrlänge führt. D.h. bei der Berechnung dieses äquivalenten U-Werts müssen Zylinderkoordinaten berücksichtigt werden. Der tatsächlichen Wärmestrom von Fluid zu Umgebung wird noch durch Übergangskoeffizienten (siehe u.A. Abschnitt Section 2.17.4.1) beinflusst.

2.17.4. Komponentendefinitionen

Eine `HydraulicNetworkComponent` definiert die Basiseigenschaften eines Strömungselements. Diese werden in dem Katalog `Components` mit eindeutigen IDs aufgelistet.

Beispiel 56. Definition einer Komponente

```
<Components>
    <HydraulicNetworkComponent id="1" modelType="ConstantPressurePump">
      <IBK:Parameter name="PressureHead" unit="Pa">1000</IBK:Parameter>
      <IBK:Parameter name="Volume" unit="m3">0.01</IBK:Parameter>
    </HydraulicNetworkComponent>

    ...
</Components>
```

Tabelle 36. Attribute

Attribut	Beschreibung	Format	Verwendung
`id`	Eindeutige Identifikationsnummer der Komponente	> 0	*benötigt*
`modelType`	Modelltyp	string	*benötigt*

Die Parameter und Attribute sind dann abhängig vom `modelType` der Komponente und dem `modelType` des Netzwerks.

Im thermischen Modell kann für jeden Modelltyp noch ein (optionales) Wärmeaustauschmodell definiert werden (siehe Section 2.17.6).

Modelltyp: SimplePipe

`SimplePipe` ist ein einfaches Rohrmodell, bei dem das gesamte Rohr als ein zusammenhängendes Fluidvolumen mit entsprechend gemittelten Eigenschaften beschrieben wird.

Für das Model `SimplePipe` werden keine weiteren Parameter benötigt.

Modelltyp: DynamicPipe

Die `DynamicPipe` ist ein detailliertes Rohrmodell, bei dem das Rohr entlang der Rohrlänge räumlich diskretisiert wird.

Es werden die folgenden Parameter benötigt

Name	Einheit	Beschreibung	Werteberei ch	Verwendung
`PipeMaxDiscretizationWidth`	m	Länge der diskretisierten Elemente	>0	*benötigt*

Modelltyp: ConstantPressurePump

Für das Model `ConstantPressurePump` werden diese Parameter benötigt:

Name	Einheit	Beschreibung	Werteberei ch	Verwendung
`PressureHead`	Pa	Konstante Druckhöhe, welche die Pumpe erzeugt	beliebig	
`PumpEfficiency`	-	Gesamtwirkungsgrad der Pumpe	0...1, > 0.0	*benötigt für Modelltyp* `ThermalHydraulicNetwork`
`Volume`	m3	Fluid volume inside the pump	> 0.0	*benötigt für Modelltyp* `ThermalHydraulicNetwork`

Die Pumpeneffizienz ist als der mechanische Gesamtwirkungsgrad der Pumpe definiert. D.h. die durch Volumenstrom und Druckhöhe gegebene mechanische Arbeit entspricht diesem Anteil der Gesamtarbeit. Die Differenz der Leistungen wird als Wärmequelle dem Fluid aufgeprägt.

ModellTyp: HeatExchanger

Das Model `HeatExchanger` ist ein einfacher Wärmeübertrager, welcher mit dem Fluid einen vorgegebenen Wärmestrom austauscht. Es werden diese Parameter benötigt:

Name	Einheit	Beschreibung	Werteberei ch	Verwendung
`HydraulicDiameter`	mm	Äquivalenter hydraulischer Durchmesser (wird für die Berechnung des Strömungsquerschnitts und der Strömungsgeschwindigkeit benötigt)	> 0.0	*benötigt*
`PressureLossCoefficient`	---	Effektiver Druckverlustbeiwert	> 0.0	*benötigt*
`Volume`	m3	Fluidvolumen im Wärmetauscher	> 0.0	*benötigt für Modelltyp* `ThermalHydraulicNetwork`

2.17.5. Strömungselemente

Das eigentliche Netzwerk wird durch die Definition konkreter Strömungselemente aufgebaut. Diese sind untereinander durch Einlass- und Auslassknoten verknüpft.

Die tatsächlichen Strömungselemente des Netzwerks werden innerhalb des XML-tags `Elements` mit dem XML-tag `HydraulicNetworkElement` definiert.

Beispiel 57. Definition der verknüpften Strömungselemente (Knotennummerierung erfolgt implizit)

```
<Elements>
  <HydraulicNetworkElement id="1" inletNodeId="5" outletNodeId="6" componentId="1" pipePropertiesId="1">
    <IBK:Parameter name="Length" unit="m">100</IBK:Parameter>
  </HydraulicNetworkElement>
  <HydraulicNetworkElement id="2" inletNodeId="6" outletNodeId="7" componentId="2">
  </HydraulicNetworkElement>
  ...
</Elements>
```

`HydraulicNetworkElement`-tags haben die folgenden Attribute:

Tabelle 37. Attribute

Attribut	Beschreibung	Format	Verwendung
`id`	Eindeutige Identifikationsnummer des Strömungselements	> 0	*benötigt*
`displayName`	Anzeigename/Beschreibung (verwendet für Ausgaben)	string	optional
`inletNodeId`	ID des Einlassknotens	> 0	*benötigt*
`outletNodeId`	ID des Einlassknotens	> 0	*benötigt*
`componentId`	ID des referenzierten `HydraulicNetworkComponent`	> 0	*benötigt*
`pipePropertiesId`	ID des referenzierten `HydraulicNetworkPipeProperties`	> 0	optional (*benötigt für Rohre*)

Die ID eines `HydraulicNetworkElement` muss global eindeutig sein, d.h. Strömungselemente müssen netzwerkübergreifend mit einer eindeutigen ID bezeichnet werden. Komponenten-IDs/Rohreigenschaften-IDs müssen nur innerhalb eines Netzwerkes eindeutig sein.

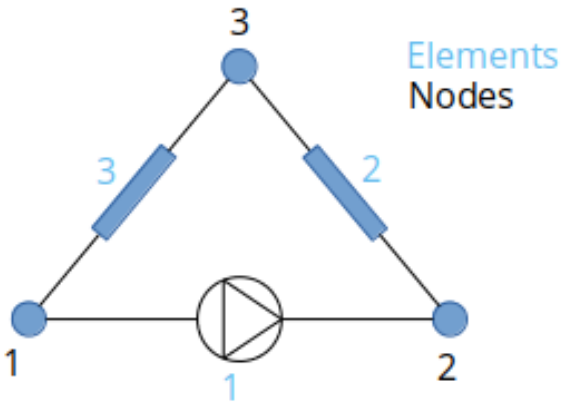


Abbildung 12. Einfaches Strömungsnetzwerk mit 3 Knoten und 3 Elementen

Die Strömungselemente sind miteinander durch Knoten verknüpft. In jedem Strömungselement fließt das Fluid (geplant) von dem Knoten mit der `inletNodeId` zu dem Knoten mit der `outletNodeId`. Während der Berechnung ist es jedoch möglich, dass sich der Massestrom umkehrt. Dies ändert aber nichts an der Topologiedefinition des

Netzwerkes. Man könnte `inletNodeId` auch mit "Knoten 1 des Elements" und `outletNodeId` mit "Knoten 2 des Elements" bezeichnen.

Abbildung 12 zeigt ein einfaches Netzwerk bestehend aus 3 Elementen. Ein solches Netzwerk würde wie folgt definiert werden (Beispiel 58).

Beispiel 58. XML-Definition eines einfachen Strömungsnetzwerks mit 3 Knoten und 3 Elementen

```
<Elements>
    <!-- Pump -->
  <HydraulicNetworkElement id="1" inletNodeId="1" outletNodeId="2" componentId="1"/>
  <!-- Pipe id=2-->
  <HydraulicNetworkElement id="2" inletNodeId="2" outletNodeId="3" componentId="2" pipePropertiesId="1">
    <IBK:Parameter name="Length" unit="m">10</IBK:Parameter>
  </HydraulicNetworkElement>
  <!-- Pipe id=3-->
    <HydraulicNetworkElement id="3" inletNodeId="3" outletNodeId="1" componentId="2" pipePropertiesId="1">
    <IBK:Parameter name="Length" unit="m">6</IBK:Parameter>
  </HydraulicNetworkElement>
</Elements>
```

Verschiedene Strömungselemente sind durch die Knoten IDs `inletNodeId` und `outletNodeId` verknüpft. Die Knoten-IDs referenzieren keine Strömungselemente, sondern "virtuelle" Knoten.

Jedes Strömungselement referenziert jeweils eine Komponente mit der `componentId`.

Rohr-Elemente

Ist eine Komponente ein Rohr (z.B. `DynamicPipe`), **müssen** entsprechende Rohrparameter mit der `pipePropertiesId` referenziert werden.

Weiterhin **muss** für ein Rohrelement der Parameter `Length` definiert werden (siehe auch Beispiel 59):

Name	Einheit	Beschreibung	Wertebereich	Verwendung
`Length`	m	Rohrlänge	> 0.0	*benötigt*

Beispiel 59. Definition eines Rohrelements

```
<HydraulicNetworkElement id="2" inletNodeId="0" outletNodeId="1" componentId="3" pipePropertiesId="1">
  <IBK:Parameter name="Length" unit="m">100</IBK:Parameter>
</HydraulicNetworkElement>
```

Parallele Rohrregister (Flächenheizungen/Fußbodenheizung)

Durch Angabe des `IBK:IntPara` Tags mit dem Namen `NumberParallelPipes` kann man ein Rohrregister definieren und so z.B. eine Flächenheizung/Fußbodenheizung modellieren. Dabei strömt letztlich das Fluid parallel durch die angebene Anzahl gleichartiger Rohre.

Beispiel 60. Definition eines Rohrregisters

```
<HydraulicNetworkElement id="101" inletNodeId="2" outletNodeId="1" componentId="3" pipePropertiesId="1">
  <IBK:Parameter name="Length" unit="m">100</IBK:Parameter>
  <!-- We have a pipe register here, consisting of 10 parallel pipes -->
  <IBK:IntPara name="NumberParallelPipes">10</IBK:IntPara>
</HydraulicNetworkElement>
```

Die Ausgabe Massestrom `FluidMassFlux` und Volumenstrom `FluidVolumeFlow` beziehen sich auf das gesamte Rohrbündel.

2.17.6. Definition der Wärmeaustauschmodells (HeatExchangeType)

Für thermische Netzwerken kann für jedes Strömungselement (Section 2.17.5) ein Wärmeaustausch definiert werden. Dafür muss innerhalb der Definition des Strömungselements ein XML-Element `HeatExchangeType` definiert werden.

Beispiel 61. Definition von Strömungselementen mit `HeatExchangeType`

```
<!-- Heat exchange with heat loss/gain pre-defined in time series -->
<HydraulicNetworkElement id="1" inletNodeId="1" outletNodeId="2" componentId="2" displayName="heat exchanger">
    <!-- Definition of pre-defined heat loss -->
  <HydraulicNetworkHeatExchange modelType="HeatLossSpline">
    <LinearSplineParameter name="HeatLoss" interpolationMethod="linear">
      <TSVFile>${Project Directory}/climate/HeatFlux.csv?2</TSVFile>
    </LinearSplineParameter>
  </HydraulicNetworkHeatExchange>
</HydraulicNetworkElement>

<!-- Pipe with heat exchange to the environment (with constant environment temperature) -->
<HydraulicNetworkElement id="2" inletNodeId="2" outletNodeId="3" componentId="3" pipePropertiesId="1"
displayName="pipe">
  <IBK:Parameter name="Length" unit="m">100</IBK:Parameter>
    <!-- Definition of heat exchange with environment -->
  <HydraulicNetworkHeatExchange modelType="TemperatureConstant">
    <IBK:Parameter name="ExternalHeatTransferCoefficient" unit="W/m2K">5</IBK:Parameter>
    <IBK:Parameter name="Temperature" unit="C">0</IBK:Parameter>
  </HydraulicNetworkHeatExchange>
</HydraulicNetworkElement>
```

Der `HeatExchangeType` kann folgende Werte haben:

HeatExchangeType	Beschreibung	Verwendbar für Modelltyp
`TemperatureConstant`	Wärmeaustausch basierend auf Temperaturunterschied zwischen Medium und Umgebungstemperatur; Konstante Umgebungstemperatur ist als Parameter im `HydraulicNetworkElement` gegeben. Es muss zusätzlich der Parameter `ExternalHeatTransferCoefficient` gegeben sein.	`SimplePipe`, `DynamicPipe`

HeatExchangeType	Beschreibung	Verwendbar für Modelltyp
TemperatureSpline	Wärmeaustausch basierend auf Temperaturunterschied zwischen Medium und Umgebungstemperatur; Umgebungstemperatur ist als Zeitreihe in einem LinearSplineParameter (Section 2.2.5) gegeben. Es muss zusätzlich der Parameter ExternalHeatTransferCoefficient gegeben sein.	SimplePipe, DynamicPipe
TemperatureZone	Wärmeaustausch mit einer Zone; Raumlufttemperatur wird als Umgebungstemperatur verwendet.	SimplePipe, DynamicPipe
TemperatureConstructionLayer	Wärmeaustausch mit einer Konstruktionsschicht (Fußbodenheizung/Flächenheizung); Schichttemperatur wird als Umgebungstemperatur verwendet.	SimplePipe, DynamicPipe
HeatLossConstant	Konstanter Wärmestrom (positiv aus dem Element) ist als konstanter Parameter gegeben	SimplePipe, DynamicPipe, HeatExchanger
HeatLossSpline	Wärmestrom (positiv aus dem Element) ist als Zeitreihe in einem LinearSplineParameter (Section 2.2.5) gegeben	SimplePipe, DynamicPipe, HeatExchanger
TemperatureFMUInterface	Temperatur wird von FMU gegeben	

Wenn des XML-Element HeatExchangeType fehlt, wird die entsprechende Komponente als adiabat behandelt und verlangt entsprechend auch keine weiteren Parameter.

Parameter für Wärmeaustauschdefinition

Konstante Parameter (Tag: IBK:Parameter):

Name	Einheit	Beschreibung	Werteberei ch	Verwendung
Temperature	K	konstante Temperatur	> -200.0 ° C	TemperatureConstant
ExternalHeatTransferCoefficient	W/m2K	Äußerer Wärmeübergangskoeffizient	>= 0.0	TemperatureConstant, TemperatureSpline, TemperatureZone
HeatLoss	W	konstante Wärmeabgabe	---	HeatLossConstant

Spline-Parameter (Tag: LinearSplineParameter):

Name	Einheit	Beschreibung	Werteberei ch	Verwendung
Temperature	K	Zeitreihe mit Temperaturen	---	TemperatureSpline
HeatLoss	W	Zeitreihe mit Wärmeverlustströmen	---	HeatLossSpline

ID-Referenzen:

XML-Tag-Name	Beschreibung	Verweis auf Datentyp	Verwendung
ZoneId	Referenz zur Zone	Zone	TemperatureZone
ConstructionInstanceId	Referenz zu beheizter/gekühlter Konstruktion	ConstructionInstance	TemperatureConstructionLayer

2.17.7. Ausgaben

Die Ergebnisgrößen eines thermo-hydraulischen Netzwerkmodells werden wie folgt definiert. Als Referenzierungstyp dient entweder Network für Ausgaben des Netzwerks insgesamt, oder NetworkElement für die Adressierung individueller Strömungselemente (siehe Beispiel 62).

Beispiel 62. Objektlist für die Referenzierung eines Netzwerks mit der ID 1 und ausgewählter Elemente des Netzwerks

```
<ObjectLists>
  <ObjectList name="the Network">
    <FilterID>1</FilterID> <!-- ID of network -->
    <ReferenceType>Network</ReferenceType>
  </ObjectList>
  <ObjectList name="Pipes">
    <FilterID>1,3</FilterID> <!-- IDs of flow elements -->
    <ReferenceType>NetworkElement</ReferenceType>
  </ObjectList>
</ObjectLists>
```

Verfügbare Ausgaben

Das Netzwerk-Objekt liefert eine Vielzahl von Ergebnisgrößen für die einzelnen Strömungselemente.

Die Anforderungen an die Netzwerkausgaben richten sich allerdings nach der späteren Visualisierungsebene. Grundlegend ist davon auszugehen, dass im Postprozessing eine weitere Sicht erforderlich sein wird, welche neben dem Gebäude eine Auswertung der hydraulischen Netzwerke erlaubt. Um die Übersichtlichkeit zu wahren, wird diese Sicht von derjenigen des Gebäudes getrennt sein.

Die Netzwerkausgaben werden daher räumlich getrennt in eigenen Dateien mit dem Namensschema:

- network_<gridname>.tsv (für Ausgaben mit Referenztyp Network)
- network_elements_<gridname>.tsv (für Ausgaben mit Referenztyp NetworkElement)

angelegt. Wie bei regulären Ausgaben (siehe Section 2.14.3.2) wird der Suffix _<gridname> weggelassen, wenn es nur eine Ausgabedatei mit einem Ausgaberaster gibt.

Für Analyse der Netzwerke und Übergabesysteme sind sowohl die Masseströme und Temperaturen im Innere eines Verbindungselementes, aber auch an den Verbindungsstellen zwischen zwei Elementen von Interesse. Letzerer Fall ist beispielsweise typisch für gekoppelte Erzeuger- und Verbraucherkreisläufe, wobei eine Kontrolle der Zulauf- und Rücklauftemperatur möglich sein muss.

Da die Netzwerkvisualisierungsebene keine Knoten kennt, müssen Knotentemperaturen am Ein- und Auslass des Verbindungselementes abgegriffen werden. Ein- und Auslässe sind unabhängig von der Strömungsrichtung

entsprechend der Netzwerktopologie definiert.

Es wird bei der Topologiedefinition eines Netzwerks mittels der `HydraulicNetworkElement` tags von einer nominalen Strömungsrichtung ausgegangen. Deshalb werden Einlass- und Auslassknoten mittels der IDs `inletNodeId` und `outletNodeId` referenziert.

Je nach Bedingungen im Netzwerk ist es jedoch auch möglich, dass sich die Strömungsrichtung umkehrt, und das Medium nun auf der Einströmseite eines Rohres ausströmt. Dies wirkt sich zwar im Vorzeichen des Massestroms aus, jedoch nicht in der Bezeichung der *geometrischen* Ein- und Auslässe eines Strömungselements.

Mächte man alle Knotendrücke oder Knotentemperaturen erhalten, so kann man einfach von allen Strömungselementen die Drück am Auslass erfragen. Darüber erhält man dann alle Drücke an den jeweiligen Knoten.

Ausgaben der hydraulischen Netzwerkberechnung

Für jedes Strömungselement kann ein Massestrom ausgegen werden, wobei die Strömungsrichtung immer von *inletNode* zu *outletNode* positiv definiert ist. Der Massestrom kann über die Größe `FluidMassFlux` (in kg/s) abgefragt werden (Referenztyp `NetworkElement`).

Ebenso sind für jedes Strömungselement die Drücke am Ein- und Auslass abrufbar:

- `InletNodePressure` in Pa
- `OutletNodePressure` in Pa

Ausgaben der thermo-hydraulischen Berechnung

Jedes Strömungselement hat eine (mittlere) Temperatur, welche über die Ausgabegröße `FluidTemperature` abgefragt werden kann (Referenztyp `NetworkElement`).

Die mittlere Temperatur einen Strömungselements kann zur Visualisierung/Farbgebung des Elements verwendet werden.

Je nach physikalischer Modellierung eines Strömungselements muss die Mitteltemperatur einen Strömungselements nicht mit der Auslasstemperatur übereinstimmen (siehe Modelldokumentation). Beispiele dafür sind Speicher oder lange verlustbehaftete Rohre.

Die Temperaturen am Ein- bzw. Auslass sind (wie die Drücke) an den physischen Positionen *inletNode* und *outletNode* definiert und können ausgegeben werden. Es sind folgende Ausgabevariablen für den Referenztyp `NetworkElement` definiert:

- `InletNodeTemperature` in C
- `OutletNodeTemperature` in C
- `FlowElementHeatLoss` in W - Wärmestrom abgegeben vom Strömungselement (Energie wird dem Fluid in diesem Element entzogen). Positive Werte bedeuten Abkühlen des Mediums (Wärmeverlust).

Übersicht über Element-Ausgabegrößen:

```
ElectricalPower                 [W]         Requested electrical power for current working point
FlowElementHeatLoss             [W]         Heat flux from flow element into environment
FluidMassFlux                   [kg/s]      Fluid mass flux through a flow element
FluidTemperature                [C]         Internal fluid temperature of network element
FluidVelocity                   [m/s]       Fluid velocity
FluidViscosity                  [m2/s]      Fluid dynamic viscosity
FluidVolumeFlow                 [m3/h]      Fluid Volume flow
InletNodePressure               [Pa]        Fluid pressure at inlet node of a flow element
InletNodeTemperature            [C]         Inlet node temperature of a flow element
MechanicalPower                 [W]         Mechanical power for current working point
Nusselt                         [---]       Nusselt number
OutletNodePressure              [Pa]        Fluid pressure at outlet node of a flow element
OutletNodeTemperature           [C]         Outlet node temperature of a flow element
Prandtl                         [---]       Prandtl number
PressureDifference              [Pa]        Difference between pressures at outlet and inlet nodes
Reynolds                        [---]       Reynolds number
ThermalTransmittance            [W/K]       Total thermal transmittance of fluid and pipe wall
```

Die Auswahl einzelner Elemente via ID kann über Objektlisten recht flexibel erfolgen.

Ausgaben des Netzwerks

Zur Vereinfachung gibt es Variablen, welche für ein gesamtes Netzwerk abgerufen werden können (Referenztyp `Network`). Diese enthalten jeweils Ausgaben für alle Strömungselemente.

- `FluidMassFluxes` - Masseströme durch alle Strömungselemente des Netzwerks
- `FluidTemperatures` - Mittlere Temperaturen alle Strömungselemente des Netzwerks

Die Variablen sind vektor-wertige Größen und es muss der *Index* des jeweils angeforderten Vektorelements verwendet werden. Die Indizierung entspricht der Reihenfolge der `HydraulicNetworkElement` Tags. Beispiel 63 zeigt die Definition einer indexbasierten Ausgabe für Masseströme des Netzwerks.

Beispiel 63. Beispiel für Ausgabedefinitionen mit Network als Referenztyp

```
<!-- Outputs go to file 'network.tsv' -->
<OutputDefinition>
  <!-- We choose the flow through the second element (pipe 101) as reference flux
     for the entire network -->
  <Quantity>FluidMassFluxes[1]</Quantity>
  <ObjectListName>Entire network</ObjectListName>
  <GridName>hourly</GridName>
</OutputDefinition>
```

Übersicht über Netzwerk-Ausgabegrößen:

```
FluidMassFluxes(id,4)                   [kg/s]      Fluid mass flux trough all flow elements
NetworkZoneHeatLoad(id,1)               [W]         Complete Heat load to zones from all hydraulic network elements
NetworkActiveLayerHeatLoad(id,1)        [W]         Heat load to the construction layers from all hydraulic network
elements
```

Variablennamen in Ausgabedateien

Variablen für Ausgaben vom Referenztyp `NetworkElement` werden in den Ausgabedateien wie folgt angegeben: `NetworkElement(id=1).FluidMassFlux` wobei hier `id=1` die ID des ausgewählten Netzwerkelements.

TODO : Datei `VariableSubstitutions.txt` beschreiben.

---

3. Referenz

3.1. Einheitendefinitionen

Im gesamten NANDRAD-Solver werden Einheiten *nur* für Ein-/ Ausgabezwecke verwendet. Innerhalb der Berechnungsfunktionen werden *immer* die SI-Basiseinheiten verwendet, wodurch Probleme durch Einheitenumrechnungen vermieden werden.

Das Einheitensystem in NANDRAD verwendet die Konvention, dass maximal ein / (Schrägstrich) Teil des Einheitennamens sein darf. Alle Einheiten, die dem Schrägstrich folgen, stehen im Nenner der Einheit. Exponenten stehen nur hinter der Einheit, zum Beispiel `m2`. Punkte in Exponenten werden weggelassen, z. B. `h05` für die Quadratwurzel der Stunde. Mehrere Einheiten werden einfach ohne . oder *-Zeichen aneinandergereiht, z. B. `kWh` oder `kg/m2s`.

Bei Einheiten wird zwischen Groß- und Kleinschreibung unterschieden! Zum Beispiel ist `Deg` korrekt, während `deg` nicht als korrekte Einheit erkannt wird.

SI-Basiseinheit	Konvertierbare Einheiten
-	
---	%, 1
---/d	%/d
1/K	
1/logcm	
1/m	1/cm
1/Pa	
1/s	1/min, 1/h
J	kJ, MJ, MWh, kWh, Wh
J/K	kJ/K
J/kg	kJ/kg
J/kgK	kJ/kgK, Ws/kgK, J/gK, Ws/gK
J/m2	kJ/m2, MJ/m2, GJ/m2, J/dm2, J/cm2, kWh/m2
J/m2s	W/m2, kW/m2, MW/m2, W/dm2, W/cm2

SI-Basiseinheit	Konvertierbare Einheiten
J/m3	Ws/m3, kJ/m3, MJ/m3, GJ/m3, J/dm3, J/cm3, kWh/m3
J/m3K	kJ/m3K
J/m3s	kJ/m3s, MJ/m3s, J/dm3s, J/cm3s, J/m3h, W/m3, kW/m3, MW/m3, W/dm3, W/cm3, W/mm3
J/mol	kJ/mol
J/s	J/h, J/d, kJ/d, W, kW, MW, Nm/s
K	C
K/m	
K/Pa	
kg	g, mg
kg/kg	g/kg, mg/kg
kg/m	g/m, g/mm, kg/mm
kg/m2	kg/dm2, g/dm2, g/cm2, mg/m2
kg/m2s	g/m2s, g/m2h, g/m2d, kg/m2h, mg/m2s, µg/m2s, mg/m2h, µg/m2h
kg/m2s05	kg/m2h05
kg/m3	kg/dm3, g/dm3, g/cm3, g/m3, mg/m3, µg/m3, log(kg/m3), log(g/m3), log(mg/m3), log(µg/m3)
kg/m3s	g/m3s, g/m3h, kg/m3h, mg/m3s, µg/m3s, mg/m3h, µg/m3h
kg/m3sK	g/m3sK, g/m3hK, kg/m3hK, mg/m3sK, µg/m3sK, mg/m3hK, µg/m3hK
kg/mol	g/mol
kg/ms	
kg/s	kg/h, kg/d, g/d, g/a, mg/s, µg/s
kWh/a	
kWh/m2a	
l/m2s	l/m2h, l/m2d, mm/d, mm/h
l/m3s	l/m3h
logcm	
logm	
logPa	
Lux	kLux
m	mm, cm, dm
m/s	cm/s, cm/h, cm/d
m/s2	
m2	mm2, cm2, dm2
m2/kg	
m2/m3	
m2/s	cm2/s, m2/h, cm2/h

SI-Basiseinheit	Konvertierbare Einheiten
m2K/W	
m2s/kg	
m3	mm3, cm3, dm3
m3/m2s	m3/m2h, dm3/m2s, dm3/m2h
m3/m2sPa	m3/m2hPa
m3/m3	Vol%
m3/m3d	Vol%/d
m3/s	m3/h, dm3/s, dm3/h
m3m/m3m	m3mm/m3m
mm/m	
mol	mmol
mol/kg	mol/g
mol/m3	mol/ltr, mol/dm3, mol/cm3
Pa	hPa, kPa, Bar, PSI, Torr
Pa/m	kPa/m
Person/m2	
Rad	Deg
s	min, h, d, a, sqrt(s), sqrt(h), ms
s/m	kg/m2sPa
s/s	min/s, h/s, d/s, a/s
s2/m2	
W/K	
W/m2K	
W/m2K2	
W/m2s	W/m2h, kW/m2s, MW/m2s, W/dm2s, W/cm2s
W/mK	kW/mK
W/mK2	
W/Person	kW/Person
undefined	

Die Einheit `undefined` bedeutet *nicht initialisiert* (intern) und darf in Eingabedateien nicht verwendet werden.

3.2. Mengenreferenzen

Die folgende Liste von Größen ist eine Übersicht über alle verfügbaren Ergebnisse, die als Ausgaben angefordert werden können. Welche Ausgaben tatsächlich verfügbar sind, hängt vom Projekt ab und wird in der Datei

`var/output_reference_list.txt` ausgegeben (siehe Diskussion im Abschnitt Outputs/Ergebnisse).

Einige der Größen sind vektorwertige Größen, gekennzeichnet mit einem Suffix `(id,xxx)` oder `(index,xxx)`. Um auf diese Werte zuzugreifen, muss die id/der Index in der Ausgabedefinition angeben werden (siehe Erklärung und Beispiele im Abschnitt Outputs/Ergebnisse).

Referenz/Objekttyp	Menge	Einheit	Beschreibung
ConstructionInstance	FluxHeatConductionA	W	Wärmeleitungsfluss über die Schnittstelle A (in die Konstruktion).
ConstructionInstance	FluxHeatConductionB	W	Wärmeleitfluss über die Schnittstelle B (in die Konstruktion).
ConstructionInstance	LayerTemperature(index,xxx)	C	Mittlere Schichttemperatur für angeforderte Größen.
ConstructionInstance	SurfaceTemperatureA	C	Oberflächentemperatur an der Schnittstelle A.
ConstructionInstance	SurfaceTemperatureB	C	Oberflächentemperatur an Grenzfläche B.
Location	AirPressure	Pa	Luftdruck.
Location	Albedo	---	Albedo-Wert der Umgebung [0..1].
Location	AzimuthAngle	Deg	Solarer Azimut (0 - Nord).
Location	CO2-CO2Concentration	---	Umgebende CO2-Konzentration.
Location	CO2Density	kg/m3	Ambiente CO2-Dichte.
Location	DeclinationAngle	Deg	Solare Deklination (0 - Nord).
Location	ElevationAngle	Deg	Solare Elevation (0 - am Horizont, 90 - direkt darüber).
Location	LWSkyRadiation	W/m2	Langwellige Himmelsstrahlung.
Location	Latitude	Deg	Breitengrad.
Location	Longitude	Deg	Längengrad.
Location	MoistureDensity	kg/m3	Feuchtedichte der Umgebung.
Location	RelativeHumidity	%	Relative Feuchte.
Location	SWRadDiffuseHorizontal	W/m2	Diffuse kurzwellige Strahlungsflussdichte auf horizontaler Fläche.
Location	SWRadDirectNormal	W/m2	Direkte kurzwellige Strahlungsflussdichte in normaler Richtung.
Location	Temperature	C	Außentemperatur.
Location	VaporPressure	Pa	Umgebungs-Dampfdruck.
Location	WindDirection	Deg	Windrichtung (0 - Nord).

Referenz/Objekttyp	Menge	Einheit	Beschreibung
Location	WindVelocity	m/s	Windgeschwindigkeit.
Model	VentilationHeatFlux(id,xxx)	W	Wärmestrom durch natürliche Lüftung
Model	VentilationRate(id,xxx)	1/h	Luftwechselrate (natürliche Lüftung)
Zone	AirTemperature	C	Raumlufttemperatur.
Zone	CompleteThermalLoad	W	Summe aller Wärmeströme in den Raum und Energiequellen.
Zone	ConstructionHeatConductionLoad	W	Summe der Wärmeleitungsflüsse von Konstruktionsoberflächen in den Raum.
Zone	VentilationHeatLoad	W	Wärmelast in den Raum durch natürliche Lüftung.