ENTWICKLUNG EINES ONLINE-SIMULATIONSWERKZEUGS ZUR INTERAKTIVEN NUTZUNG VON MODELICA-BIBLIOTHEKEN ZUR ENERGETISCHEN GEBÄUDESIMULATION FÜR DEN EINSATZ IN DER LEHRE

C. Nytsch-Geusen[1], W. Kaul[1], D. Kreulitsch[1] und J. Rädler[1]
[1]Institut für Architektur und Städtebau, UdK Berlin, Berlin, Deutschland
E-Mail: nytsch@udk-berlin.de

KURZFASSUNG

Für den Einsatz von Modelica-Modellbibliotheken in der Lehre wurde ein Server-basierter Ansatz auf Basis von Jupyter-Notebooks und des Open Source-Simulationswerkzeuges OpenModelica entwickelt, welcher eine einfache Nutzung didaktisch aufbereiteter Modelica-Modelle im Web-Browser gestattet. Hierbei wird der Modelica-Quellcode hinter der Weboberfläche für den Nutzer verborgen. Das Webinterface ermöglicht die Durchführung und Auswertung des Simulationsexperiments und ist mit didaktisch wichtigen Informationen angereichert, die sich auf die jeweilige Aufgabenstellung aus der Bauphysik oder der Gebäudetechnik beziehen. Die Nutzer können dann in einfacher Weise Simulationsmodelle auswählen und deren Parameter interaktiv in einem sinnvoll begrenzten Rahmen variieren, um so das Modellverhalten zu studieren.

Der Beitrag beschreibt zum einen den didaktischen Ansatz und zum anderen seine softwaretechnische Umsetzung. Mit Hilfe von zwei Beispielen aus den Bereichen der Bauphysik und der Gebäudetechnik auf Basis der Modelica-Bibliothek BuildingSystems wird die Anwendung des Online-Simulationswerkzeugs demonstriert.

ABSTRACT

For the use of Modelica model libraries in teaching, a server-based approach based on Jupyter notebooks and the open source simulation tool OpenModelica was developed, which allows an easy use of didactically prepared Modelica models in the web browser. The Modelica source code is hidden behind the web interface for the user. The web interface enables the simulation experiment to be carried out and evaluated and is enriched with didactically important information relating to the respective task from building physics or building technology. The users can easily select simulation models and interactively vary their parameters within a reasonably limited range in order to study the model behavior.

The article describes on the one hand the didactic approach and on the other hand the software implementation. Using two examples from the fields of building physics and building technology based on the Modelica library BuildingSystems, the use of the online simulation tool is demonstrated.

EINLEITUNG

Ein Blick in die Literatur zeigt, dass für verschiedene Gebiete der bauphysikalischen und energetischen Gebäudeplanung Online-Werkzeuge entwickelt wurden, die entweder gezielt für den Einsatz in der Lehre konzipiert wurden oder aber hierfür genutzt werden können. So wurden beispielsweise innerhalb des Forschungsverbundprojekts "multimediales Lernnetz Bauphysik" für die bauphysikalische Lehre an Hochschulen und Universitäten Wissens-, Übungs- und Berechnungsmodule für die Bereiche Wärme, Feuchte und Schall entwickelt und praktisch erprobt (siehe Abbildung 1).

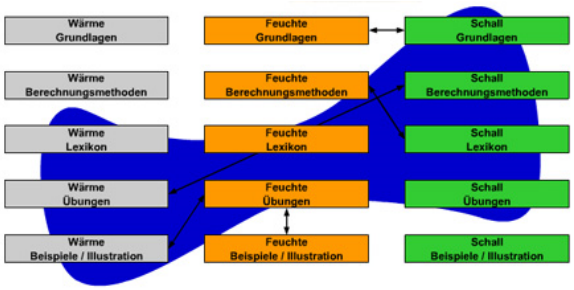

Abbildung 1: Berechnungsmodule des "Multimedialen Lernnetzes Bauphysik" (Quelle: Bauhaus-Universität Weimar)

Alle Module des „Lernnetzes Bauphysik" verfügen über ein einheitliches Interface, welches auf den Ebenen Bauelement, Einzelraum oder Gebäude von den Studierenden über ein Webportal ergänzend zu der Präsenzlehre im Selbststudium genutzt werden kann (Mehra et al. 2004).

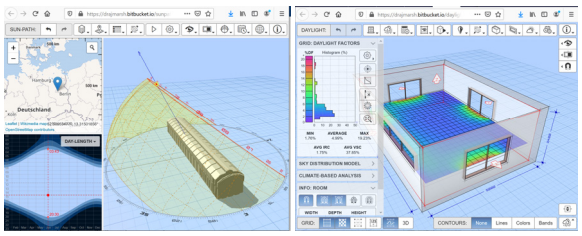

Abbildung 2: Web-Applikationen zur energetischen Gebäudeanalyse (Quelle: Marsh 2020)

In Abbildung 2 sind zwei der Web-Applikationen von Andrew Marsh dargestellt, welche für die energetische Analyse von Gebäuden intuitiv in leicht verständlicher Weise in der Lehre angewandt werden können. Die linke Abbildung zeigt ein Tool zur

 DOI: 10.3217/978-3-85125-786-1-57

Verschattungsanalyse von Baukörpern, die rechte Abbildung ein weiteres Tool zur Tageslichtanalyse von Innenräumen. In den Webtools von Marsh können beispielsweise der Gebäudestandort und die Gebäudeorientierung, der betrachtete Zeitpunkt im Jahr oder die Dimensionen des Raums und von transparenten Bauteilen interaktiv im Webbrowser durch den Nutzer verändert werden, worauf das Berechnungsergebnis unmittelbar aktualisiert wird. Die Berechnungsalgorithmen werden bei diesen Programmen als JavaScript-Bibliohek vom Server auf den lokalen Rechner des Nutzers geladen und dort ausgeführt.

Im Bereich der Simulation von Anlagen zur Energieversorgung von Gebäuden sind die Online-Varianten der Simulationsprogramme PVSol (PVSol Online 2020) und TSol (TSol Online 2020) sowie Polysun (Polysun Online 2020) erwähnenwert. Abbildung 3 zeigt beispielhaft das Webinterface von Polysun, bei dem das gewählte Anlagenschema zunächst visualisiert wird (Abbildung 3 links) und dessen Anlagenkomponenten individuell über Dialoge und Auswahlmenüs vom Nutzer parametrisiert werden können (Abildung 3 rechts). Die Durchführung der Simulationsrechnung wird in diesem Fall auf dem Server des Programmherstellers durchgeführt und danach die Ergebnisse dem Nutzer über ein pdf-File per Email zur Verfügung gestellt.

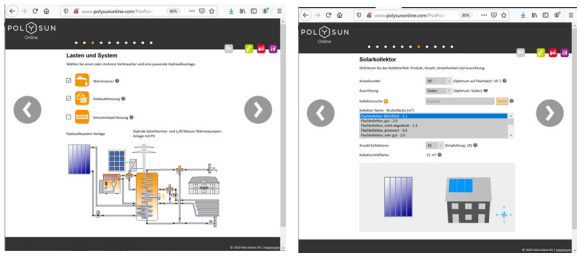

Abbildung 3: Online-Variante von Polysun

Das in diesem Beitrag beschriebene Online-Simulationswerkzeug für den Einsatz in der Lehre verfolgt einen etwas anderen Ansatz:

In einer Lehrveranstaltung soll eine gemeinsame Aufgabe oder ein Aufgabenkatalog von den Studierenden bearbeitet werden. Das Online-Simulationswerkzeug stellt dem Endnutzer die notwendigen Simulationsmodelle über einen Server zentral bereit und erlaubt eine individuelle Bearbeitung im Webbrowser. Damit kann die Reihenfolge der zu bearbeitenden Einzelaufgaben oder auch die geforderte Modellparametrierung für einzelne Kursteilnehmer von der Aufgabenstellung her unterschiedlich festgelegt werden.

Als Grundlage der Modellbildung wird die Modellierungssprache Modelica (Modelica 2020) genutzt, welche mittlerweile weltweit für die energetische Gebäudesimulation in vielen unterschiedlichen Teilbereichen von der Bauteilsimulation, der Raum- und Gebäudesimulation bis zur Quartierssimulation verwendet wird, was im internationalen Verbundprojekt IBPSA Project 1 dokumentiert wird (Wetter et al. 2019, IBPSA Project 1, 2020).

Die Bearbeitung von Modelica-Modellen und ihre Ausführung im Simulationsexperiment in einem Webbrowser wurde über die Webapplikation OMWebbook demonstriert (OMWebbook, 2020). Der Modelica-Quellcode wird hier direkt im Webbrowser editiert, das Simulationsexperiment auf einem zentralen Server auf Basis von OpenModelica ausgeführt und die Ergebnisse dem Endnutzer im im gleichen Webbrowser angezeigt (vgl. Abbildung 4).

Abbildung 4: Simulation von Modelica-Modellen mit OMWebbook

Die für die energetische Gebäude- und Anlagensimulation konzipierten Modelica-Modell-biblioheken Buildings, AixLib, IDEAS und BuildingSystems sind durch die gemeinsam verwendete Modelica IPBSA-Bibliothek (https://github.com/ibpsa/modelica-ibpsa) sehr gut untereinander kombinierbar und durch den objekt- und gleichungsbasierten Ansatz von Modelica zudem leicht erweiterbar.

Leider sind diese Modell-Bibliotheken für technisch weniger versierte Nutzer, wie z.B. Studierende im Grundstudium, durch ihre Komplexität nicht immer leicht verständlich und anwendbar. Aus diesem Grund wird der Quellcode der Modelica-Modelle im vorliegenden Ansatz dem Endnutzer zunächst vollständig verborgen. Er kann lediglich zwischen den

für die Bearbeitung der Aufgabenstellung vorbereiteten Modelica-Systemmodellen auswählen und diese dann individuell parametrisieren.

 DOI: 10.3217/978-3-85125-786-1-57

SOFTWARETECHNISCHE UMSETZUNG

Als Grundlage des Webinterfaces für den Endnutzer wird die web-basierte interaktive Jupyter-Notebook Umgebung (Jupyter 2020) verwendet, mit der Jupyter-Notebook-Dokumente mit der Dateiendung *.ipynb erstellt werden können. Ein Jupyter-Notebook-Dokument wird als ein JSON-File gespeichert, welches eine Liste aus Eingabe- und Ausgabezellen in einem Webbrowser repräsentiert, die jeweils Programmcode, Textinformationen oder grafische Plots enthalten können. Hierbei werden die einzelnen Zellen strikt der Reihenfolge nach abgearbeitet, was der Nutzer berücksichtigen muss (siehe Abbildung 5).

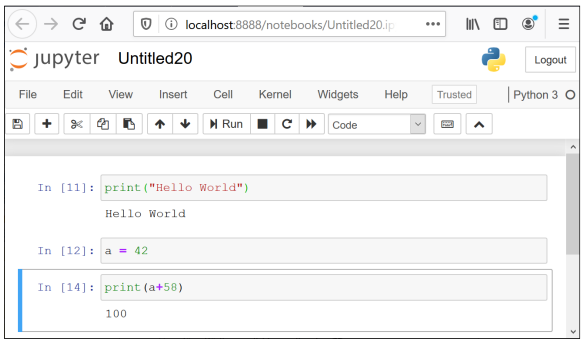

Abbildung 5: Ausgeführte Zelle in einem Jupyter-Notebook für die Sprache Python

Jupyter-Notebooks unterstützen mehr als 40 Programmiersprachen, u.a. Python, R, Julia und Scala. Im vorliegenden Fall wird die Programmiersprache Python verwendet, da diese einerseits die grundlegende Sprache zur Darstellung und Anpassung von Jupyter-Notebooks verkörpert, andererseits die meisten Modelica-Simulationstools wie Dymola, OpenModelica oder JModelica über eine Python-schnittstelle angesteuert werden können. Über die Python-Schnittstellen kann der gesamte Workflow von der Übersetzung und Parametrisierung des Modelica-Simulationsmodells bis hin zur Berechnung und Auswertung des Simulationsexperiments kontrolliert werden.

Die Standardvariante von Jupyter-Notebook erlaubt nur die Ausführung eines Notebooks von einem Anwender auf einem lokalen oder entfernten Rechner im Intra- oder Internet. Für den Fall das mehrere Nutzer gleichzeitig jeweils einen eigene Prozess, z.B. in Form von Simulationsläufen über das Jupyter-Notebook-Webinterface starten möchten, wird eine Server-Client-Lösung benötigt. Für diesen Anwendungsfall wurde vom Jupyter-Projekt die Server-Variante JupyterHub (JupyterHub 2020, siehe Softwarearchitektur in Abbildung 6) entwickelt. Die individuellen Inhalte und Bearbeitungsstände der einzelnen Nutzer werden bei JupyterHub in einer Datenbank gespeichert, wodurch eine Vielzahl von Nutzern gleichzeitig auf den Simulationsrechenkern zugreifen kann.

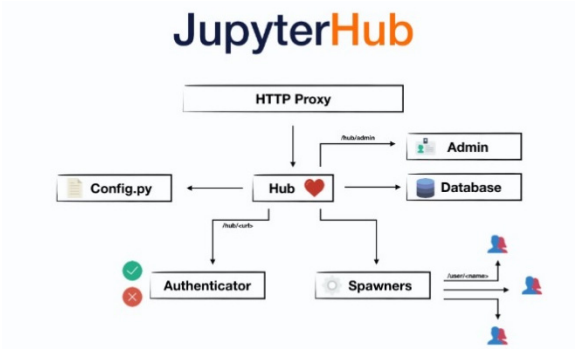

Abbildung 6: Softwarearchitektur von JupyterHub

Als Simulationsrechenkern kommt das Open Source Modelica-Werkzeug OpenModelica (OpenModelica 2020) in der Version 1.15 zur Anwendung, welches über seine Python-Schnittstelle (OMPython) mit der Jupyter-Webumgebung kommuniziert (Abbildung 7).

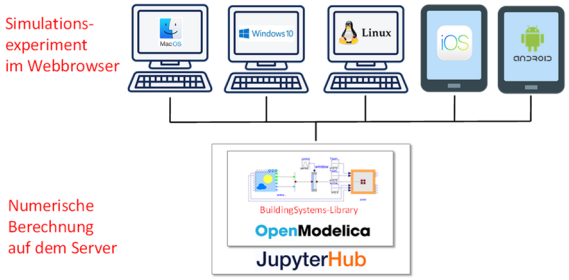

Abbildung 7: Online-Simulationswerkzeug auf Basis von JupyterHub und OpenModelica

Im vorliegenden Fall wird eine spezielle Variante von JupyterHub, „The Littlest JupyterHub“ (The Littlest JupyterHub 2020) genutzt, welche eine besonders einfache Konfiguration und Installation unterstützt und für kleinere Anwendungen von bis zu 100 Clients von den Entwicklern empfohlen wird.

ANWENDUNGSBEISPIELE

An Hand von zwei Anwendungsbeispielen, der Simulation des thermischen Verhaltens eines Wärmedämmverbundsystems sowie eines solar-thermischen Anlagensystems zur Warmwasser-bereitung, soll der Gebrauch und das Nutzerinterface des Online-Simulationswerkzeuges demonstriert werden.

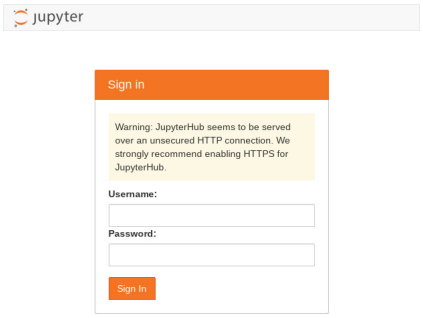

Abbildung 8: Anmeldefenster von JupyterHub

Nach Eingabe der URL erscheint das Anmeldefenster des Online-Simulationswerkzeuges, in welchem der Anwender sich zunächst mit Nutzername und

 DOI: 10.3217/978-3-85125-786-1-57

Passwort einloggt (siehe Abbildung 8). Danach kann er im Web-Browser eines der angebotenen Simulationsaufgaben über eine Startseite öffnen, wobei zwischen den Sprachversionen Deutsch oder Englisch gewählt werden kann (vgl. Abbildung 9).

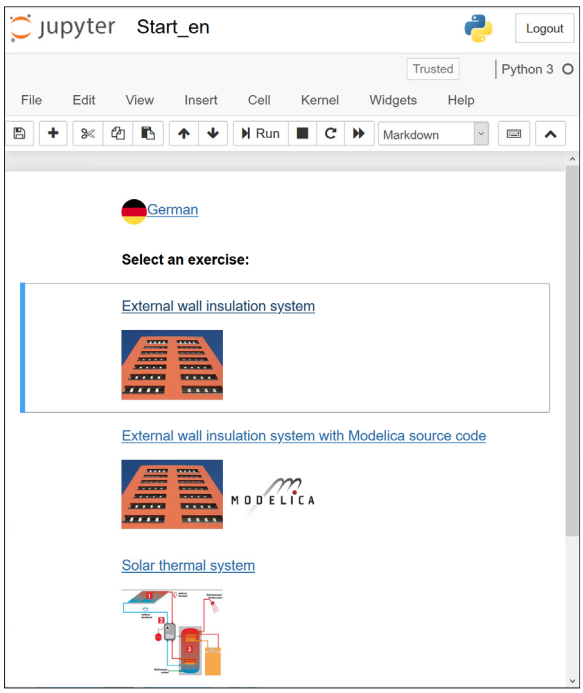

Abbildung 9: Aufgabenauswahl in der Jupyter-Notebook-Umgebung

Wärmedämmverbundsystem

In diesem Anwendungsbeispiel wird zunächst der baukonstruktive Aufbau und die Funktionsweise eines Wärmedämmverbundsystems (WDVS) erläutert (Notebook Wall_en.ipynb). Hierzu wird eine didaktische passende Kombination aus beschreibenden Text und anschaulich ergänzenden Fotos und Prinzipskizzen verwendet (vgl. Abbildung 10). Hierfür können beliebige Inhalte auf Basis der vereinfachenden Auszeichnungssprache Markdown (Markdown 2020), aber auch HTML-Elemente oder mathematische Formeln über die LaTeX-Formatierung dargestellt werden, wie sie z.B. bei der Beschreibung des physikalischen Modells in Abbildung 11 verwendet werden.

Um das dynamische Verhalten dieser mehrschichtigen opaken Baukonstruktion in einem Modell abbilden und im Simulationsexperiment analysieren zu können, kommt ein Modelica-Simulationsmodell aus der BuildingSystems-Bibliothek (Nytsch-Geusen et al. 2016) zur Anwendung. Hierin ist ein Wandmodell mit drei Schichten aus unterschiedlichen Material (Ziegel, Dämmstoff, Putz) mit einem Umgebungsmodell und einem Zonenmodell verkoppelt, um die thermischen Randbedingungen auf der Wandaußen- und Innenseite differenziert berücksichtigen zu können (siehe Abbildung 12).

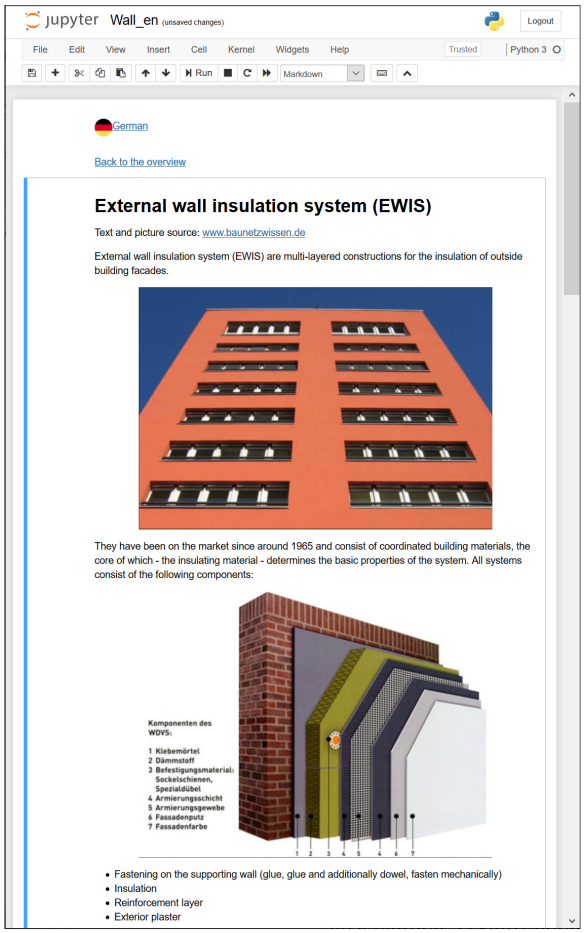

Abbildung 10: Beschreibung des Anwendungsbeispiels Wärmedämmverbundsystem

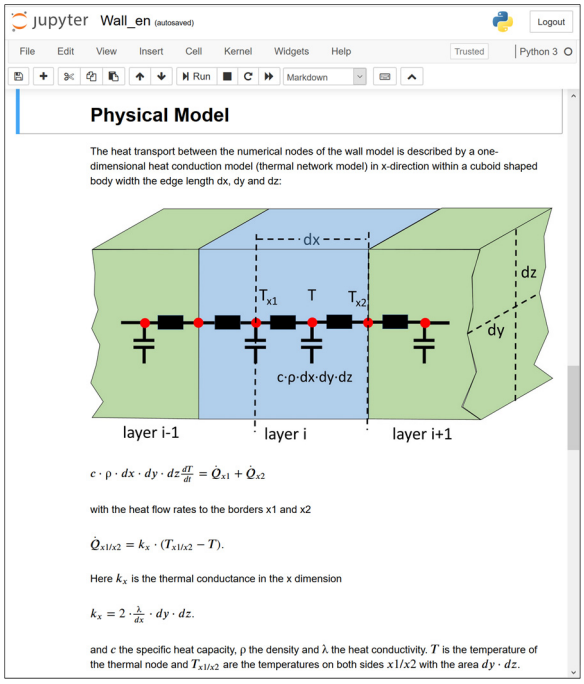

Abbildung 11: Beschreibung des physikalischen Modells, welches dem Anwendungsbeispiel Wärmedämmverbundsystem zu Grunde liegt

 DOI: 10.3217/978-3-85125-786-1-57

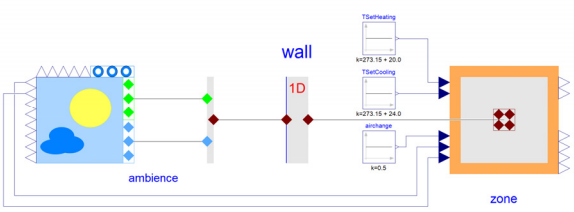

Abbildung 12: Modelica-Modell der WDVS-Fassade mit ihren klimatischen Randbedingungen

Zur Vereinfachung des Simulationsexperiments wird dieses Simulationsmodell im Jupyter-Notebook gegenüber dem Anwender nahezu vollständig verborgen.

Es können nur noch wenige Simulationsparameter in einem festgelegten Bereich (z.B. für die Ziegelschicht zwischen 12 cm bis 72 cm) und einer vorgegebenen Abstufung (12 cm Ziegelbreite) über Schieberegler angepasst werden, wodurch eine nicht sinnvolle Parameterisierung (z.B. negative oder unplausible dünne Werte für eine Ziegelschicht) ausgeschlossen werden kann (vgl. Abildung 13).

Neben den Schichtdicken der Wandkonstruktion können in diesem Beispiel folgende Parameter verändert werden:

- *Anzahl der numerischen Knoten pro Wandschicht:* Für die beiden Konstruktionsschichten „Ziegel“ und „Dämmung“ des WDVS kann die Anzahl der numerischen Knoten zwischen 1 und 10 variiert werden, um den Einfluss der Modelldiskretisierung auf das Rechenergebnis zu analysieren.
- *Ausrichtung der Wand:* Durch Festlegung des Azimutwinkels und des Neigungswinkels der Wand verändert sich entsprechend dem gewählten Standort die direkte Sonneneinstrahlung auf die Fassadenoberfläche.
- *Thermische Randbedingungen des Wandmodells*: Hier kann zwischen verschiedenen Klimastandorten (Berlin oder San Francisco) oder alternativ zwischen stationären Randbedingungen für den winterlichen Heizfall oder den sommerlichen Kühlfall gewählt werden. Wird einer der Klimastandorte gewählt, wirken sowohl die Außenluft- und Strahlungstemperatur des Himmels, die solare Einstrahlung als die Windgeschwindigkeit als Randbedingungen auf das Wandmodell. Im Fall von stationären Randbedingungen wird die solare Einstrahlung und die Windgeschwindigkeit zu null gesetzt und für die Außenluftemperatur sowie die Strahlungstemperatur des Himmels werden der gleiche konstante Wert verwendet.
- *Start- und Endzeit des Simulationsexperiments*: Hierdurch kann ein sehr kurzer Zeitraum für das Simulationsexperiment über wenige Stunden bis zu einem vollständigen Jahr festgelegt werden. Liegt das gewählte Simulationsende zeitlich vor dem gewählten Simulationsbeginn, wird der Anwender darauf hingewiesen, den Simulationszeitraum sinnvoll anzupassen.

Nach Festlegung des Simulationsexperiments kann der Prozess zur Durchführung der dynamische Simulation auf Knopfdruck gestartet werden: in der im Jupyter-Notebook hierfür zuständigen Python-Codezelle (in der Voreinstellung für den Anwender verborgen) wird OpenModelica dann über seine Python-Schnittstelle angewiesen, das passende Modelica-Modell auszuwählen, gegebenenfalls den Modellquellcode zu kompilieren und danach das Simulationsmodell zu berechnen.

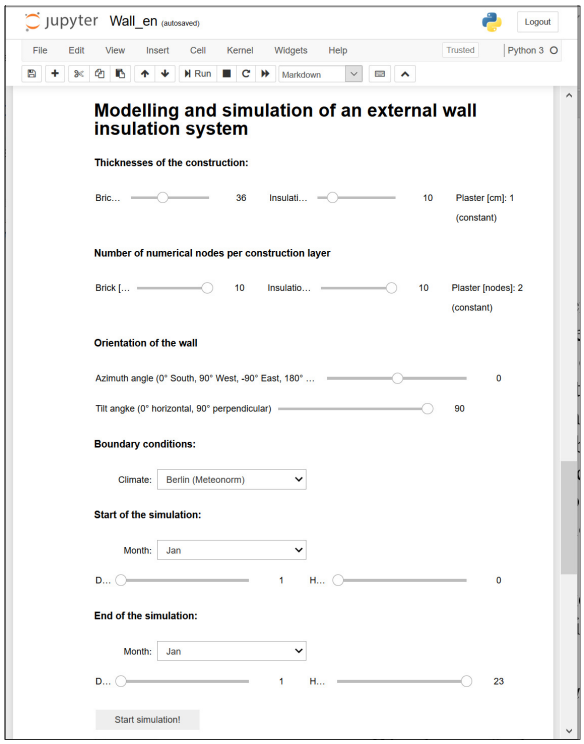

Abbildung 13: Parametrisierung des WDVS-Modells über die Nutzerschnittstelle des Jupyter-Notebooks

Dem folgenden Python-Code können die einzelnen Schritte der Modellauswahl- und Parametrisierung, der Festlegung des Simulationsexperiments sowie des Modellstarts entnommen werden:

```
def startSimulation(startTime,stopTime):
  """model selection and definition of the
     simulation experimen"""
  global mod
  if str(climateBC.value)=='constant':
    mod=ModelicaSystem("WallWDVSConstantBC",
    ["Modelica","BuildingSystems"])
  elif str(climateBC.value)=='San Francisco':
    mod=ModelicaSystem("WallWDVSSanFrancisco",
    ["Modelica","BuildingSystems"])
  else:
    mod=ModelicaSystem("WallModels.WallWDVS",
    ["Modelica","BuildingSystems"])
    mod.setParameters(
    ["wall.constructionData.thickness[1]
     ="+str(thicknessBrick.value/100),
     "wall.constructionData.thickness[2]
     ="+str(thicknessInsulation.value/100),
     "wall.constructionData.thickness[3]=0.01",
     "wall.nNodes={"+str(nodesBrick.value)+",
     "+str(nodesInsulation.value)+",2}",
     "wall.angleDegAzi=
     "+str(azimuthAngleWall.value),
```

 DOI: 10.3217/978-3-85125-786-1-57

```
      "wall.angleDegTil=
      "+str(tiltAngleWall.value)])
    mod.setSimulationOptions(
      ["startTime="+str(startTime),
       "stopTime="+str(stopTime),
       "stepSize=1800.0"])
    mod.simulate()
    out.clear_output()

def sim_button_clicked(b):
  """Simulation time period check and
     start of the similation experoment"""
  with out:
    out.clear_output()
  if stopTime > startTime:
    out.append_stdout("Simulation has started")
    startSimulation(startTime,stopTime)
    out.append_stdout("Simulation finished")
  else:
    out.append_stdout(
      "startTime is greater than stopTime ->
       please, reduce startTime !")

simButton.on_click(sim_button_clicked)
```

Nach Abschluss der Simulationsexperiments bekommt der Nutzer eine Rückmeldung über die vollständig abgeschlosssene Berechnung. Die zum Verständnis der Simulationsanalyse wesentlichen Ergebnisse werden in Form von Zeitreihen, im vorliegenden Beispiel über einen Zeitraum von 12 Tagen im Juli, visualisiert (siehe Abbildung 14).

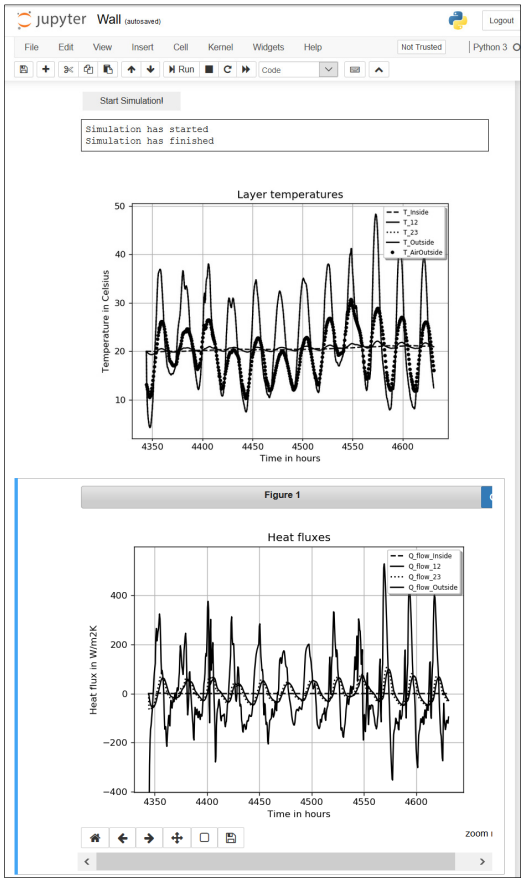

Abbildung 14: Simulierte Temperaturen (oben) und Wärmeströme (unten) an den Bauteiloberflächen sowie zwischen den Bauteilschichten der WDVS-Konstruktion

Alternativ kann im Online-Simulationswerkzeug auch eine Variante für erfahrenere Nutzer gewählt werden, welche einen direkten Zugriff auf den Quellcode der Modelica-Modelle erlaubt (vgl. Abbildung 15).

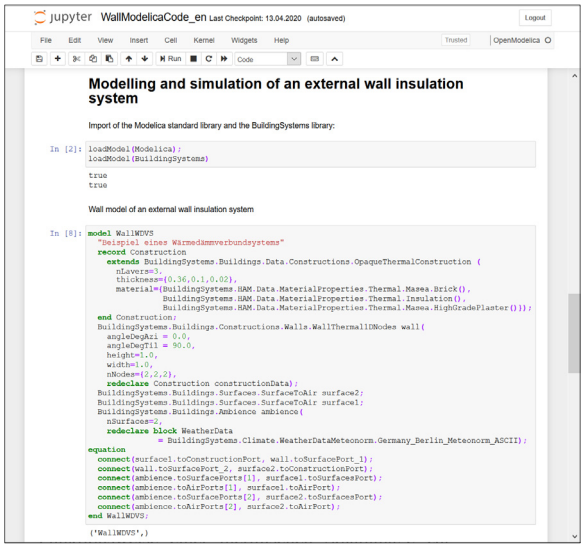

Abbildung 15: Modelica-Quellcode des WDVS im Jupyter-Notebook

Technisch wird dies über den OpenModelica-Kernel für Jupyter-Notebook realisiert (Jupyter OpenModelica, 2020), welcher das direkte Editieren, Übersetzen und Ausführen von Modelica-Modellen in Jupyter-Notebooks unterstützt.

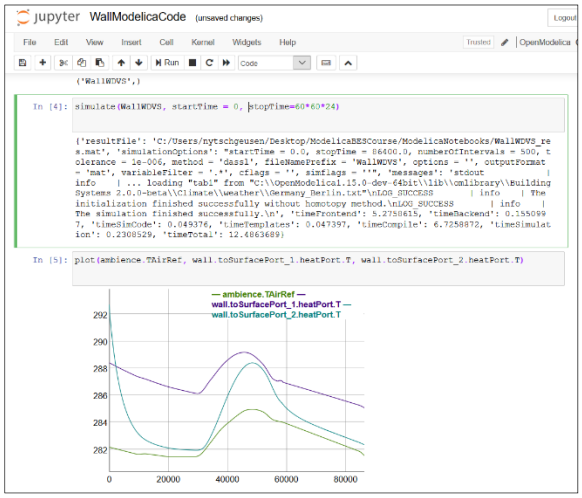

Abbildung 16: Ausführen einer Simulation durch Nutzung der OpenModelica-API in Jupyter-Notebook

Durch das Aufrufen bestimmter Funktionen der OpenModelica-API wie `simulate()` oder `plot()`, können Simulationsexperimente, definiert, gestartet bzw. Ergebnisse geplottet werden (vgl. das Jupyer-Notebook in Abbildung 16).

Thermische Solaranlage

Im zweiten Anwendungsbeispiel wird zunächst die Funktionsweise einer thermischen Solaranlage zur Warmwasserbereitung allgemein und an Hand ihrer Hauptbestandteile Solarkollektor, Solarspeicher, Wärmetauscher, Hydraulik, Nachheizsystem und

 DOI: 10.3217/978-3-85125-786-1-57

Verbraucher und ihrem Zusammenwirken als energietechnisches System über eine Skizze und erläuternden Text beschrieben (Abbildung 17).

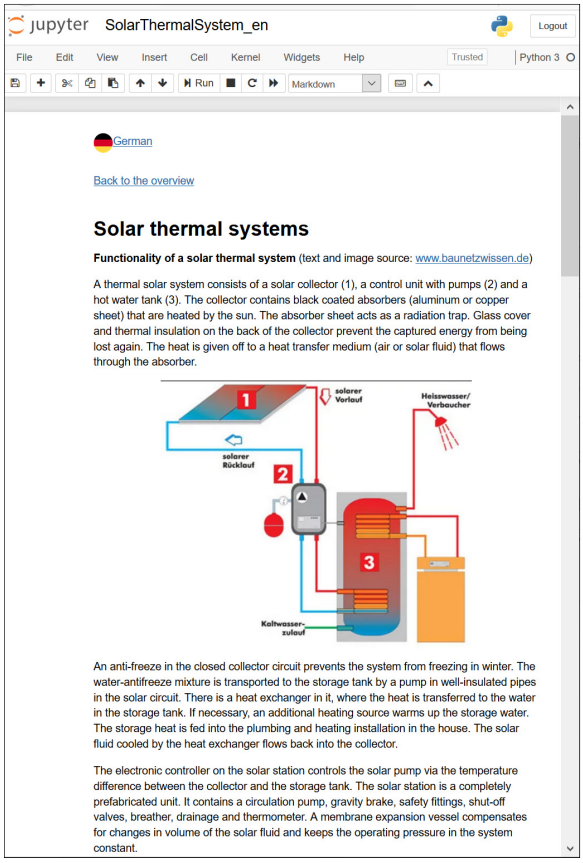

Abbildung 17: Beschreibung des Anwendungsbeispiels Thermische Solaranlage zur Warmwasserbereitung

In diesem Jupyter-Notebook-Beispiel sind die klimatischen Randbedingungen unveränderlich festgelegt (Standort in El Gouna (Ägypten) am Roten Meer). Im Webbrowser lassen sich folgende Anlagenparameter variieren (Abbildung 18):

- *Kollektorfläche*: durch Anpassung der Kollektorfläche zwischen 1 m² und 20 m² in 1 m²-Schritten, kann die Auswirkung auf die Temperaturniveaus in den Systemkomponenten Kollektor und Speicher analysiert werden.
- *Speichervolumen*: durch Anpassung des Speichervolumens zwischen 10 und 1000 Litern in 10 Liter-Schritten kann die Auswirkung auf die Temperaturschichtung im Solarspeicher untersucht werden.

Weiterhin kann der Anwender zwischen zwei verschiedenen Anlagenvarianten mit externem und internem Wärmetauscher zwischen Solarkreis und Solarspeicher wählen (vgl. die beiden Anlagenschaubilder in Abbildung 18), wobei die technischen Unterschiede zwischen beiden Systemen im einführenden Text des Anwendungsbeispiels erläutert werden.

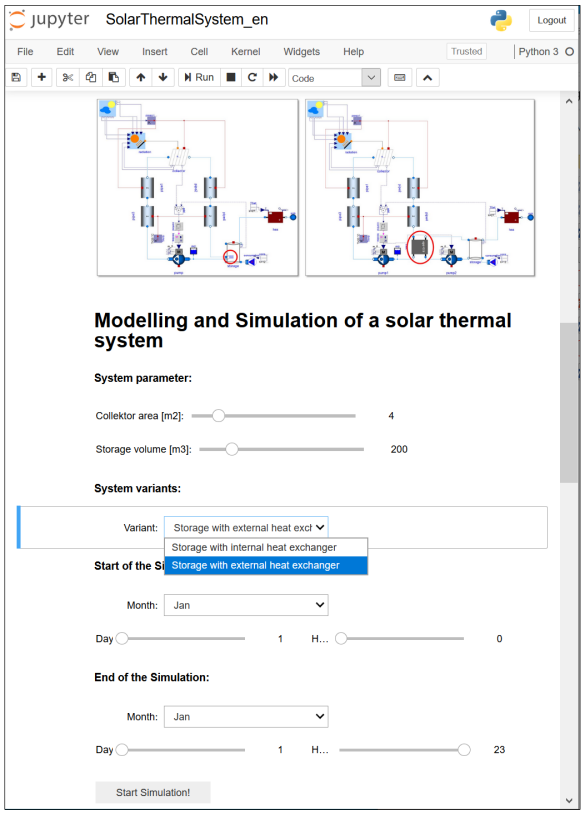

Abbildung 18: Parametrisierung des Modells der thermischen Solaranlage über die Nutzerschnittstelle des Jupyter-Notebooks

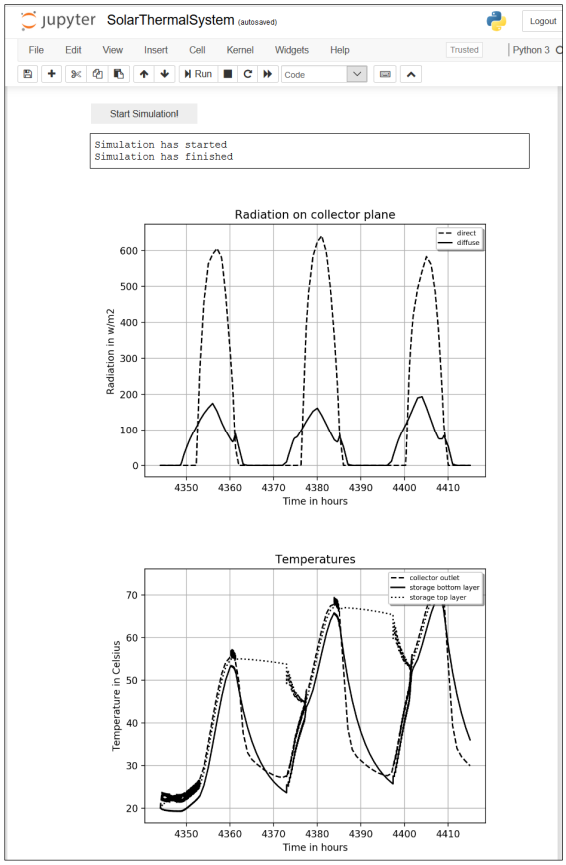

Abbildung 19: Simulationsergebnisse für die Anlagenvariante mit externem Wärmetauscher

 DOI: 10.3217/978-3-85125-786-1-57

Abbildung 19 zeigt die Simulationsergebnisse über einen Simulationszeitraum von drei Tagen im Juli für eine Anlage mit 4 m^2 Kollektorfläche und einem Speichervolumen von 290 Litern. Das obere Diagramm zeigt die direkte und diffuse Einstrahlung auf die nach Süden ausgerichtete und mit 30° geneigte Kollektorfläche, das untere Diagramm die Fluidtemperaturen im Kollektormodell sowie der obersten und untersten Fluidschicht im Solarspeichermodell.

ZUSAMMENFASSUNG

In diesem Beitrag wurde der Prototyp eines Online-Simulationswerkzeuges auf Basis von Jupyter-Notebook, JupyterHub und OpenModelica vorgestellt. Durch die Client-Server-Architektur von JupyterHub ist es möglich, das Online-Simulationswerkzeug auch für größere Übungsgruppen in Lehrveranstaltungen zu verwenden. Alle Bestandteile des Online-Simulationswerkzeuges inkl. der genutzten Modelica-Biblioheken sind als Open Source-Software frei verfügbar und ohne Hindernisse für die Lehre nutzbar.

Mit Hilfe von zwei Anwendungsbeispielen aus der Bauphysik und der Gebäudetechnik konnte gezeigt werden, dass durch entprechend aufbereitete Jupyter-Notebooks eine einfache Durch-führung von Simulationsanalysen über Auswahl-menüs und Schieberegler zur Parameteranpassung auch für wenig versierte Nutzer möglich ist, da das für das Simulationsexperiment verantwortliche Modelica-Modell auf diese Weise vollständig im Hintergrund verborgen bleibt. Durch die ergänzenden erläuternden Beschreibungen und Abbildungen in den Jupyter-Notebooks können die simulationsbasierten Aufgaben in der Lehre zudem selbsterklärend gestaltet werden.

AUSBLICK

Ausgehend von dem jetzigen Entwicklungsstand des Prototyps sollen zukünftig Erweiterungen am Nutzerinterface vorgenommen werden, was vor allem eine umfangreichere Möglichkeit zur Modell-parametrisierung betrifft. In einen nächsten Schritt sollen Simulationskurse für Fragestellungen der Bauphysik und der energetischen Gebäudetechnik entwickelt werden, welche jeweils eine größere Anzahl aufeinander aufbauender Jupyter-Notebooks enthalten. Diese sollen dann in Lehrveranstaltungen für Studierende der Architektur und der Energietechnik in der praktischen Anwendung evaluiert werden. Für diesen Zweck sollen auch die Möglichkeiten des Postprocessing erweitert werden, um neben der Darstellung von Zeitreihen auch kumulierte oder durchschnittliche Größen, wie z.B. monatliche Energiemengen oder die solare Deckungsrate eines solarthermischen Systems über einen Zeitraum visualisieren zu können.

LITERATUR

IBPSA Project 1, 2020. Webpage IBPSA Project 1. BIM/GIS and Modelica Framework for building and community energy system design and operation: https://ibpsa.github.io/project1, letzter Zugriff 2.4.2020.

Jupyter Notebook 2020. Webpage Jupyter-Projekt: https://jupyter.org, letzter Zugriff 2.4.2020.

Jupyter OpenModelica 2020. GitHub-Repository https://github.com/OpenModelica/jupyter-openmodelica, letzter Zugriff 2.4.2020.

JupyterHub, 2020. Webpage JupyterHub-Projekt: https://jupyterhub.readthedocs.io, letzter Zugriff 2.4.2020.

Mehra, S.-R., Sedlbauer, K. 2004. Innovative Lehr- und Lernmethoden in der bauphysikalischen Ausbildung. Bauphysik. 26. 385 - 391. 10.1002/bapi.200490100.

Markdown, 2020. Webpage Mardown-Spezifikation: https://daringfireball.net/projects/markdown, letzter Zugriff 2.4.2020.

Marsh, A. 2020: Web Applications of Andrew Marsh http://andrewmarsh.com/software/#applications, letzter Zugriff 2.4.2020.

Modelica 2020. Webpage Modelica-Association: https://modelica.org, letzter Zugriff 2.4.2020.

Nytsch-Geusen, C., Banhardt, C., Inderfurth., A., Mucha, K., Möckel, J., Rädler, J., Thorade, M., Tugores, C. 2016. BuildingSystems – Eine modular hierarchische Modell-Bibliothek zur energetischen Gebäude- und Anlagensimulation. Proceedings BAUSIM 2016. Dresden.

OMWebBook 2020. Webpage OMWebBook: http://omwebbook.openmodelica.org, letzter Zugriff 2.4.2020.

OpenModelica 2020. Webpage OpenModelica-Projekt: https://openmodelica.org, letzter Zugriff 2.4.2020.

Polysun Online 2020: Webportal Online-Version Polysun: http://www.polysunonline.com, letzter Zugriff 2.4.2020.

PVSol Online 2020: Webportal Online-Version PVSol:http://pvsol-online.valentin-software.com, letzter Zugriff 2.4.2020.

The Littlest JupyterHub 2020. Webpage The Littlest JupyterHub-Projekt: http://tljh.jupyter.org, letzter Zugriff 2.4.2020.

TSol Online 2020: Webportal Online-Version TSol: http://valentin.de/calculation/thermal/system/ww/de, letzter Zugriff 2.4.2020.

Wetter, M., van Treeck, C., Helsen, L., Maccarini, A., Saelens, D., Robinson, D., Schweiger, G. 2019. IBPSA, Project 1: BIM/GIS and Modelica framework for building and community energy system design and operation–ongoing developments, lessons learned and challenges. IOP Conference Series: Earth and Environmental Science.

 DOI: 10.3217/978-3-85125-786-1-57