1. Demonstration of Engineering Model (EM) Priming with Python

The following is a demonstrative python code that users may use to simulate corn production. The program uses Python Crop Simulation Environment [1] which is one the standard crop simulation environments. All the code in Green color is part of regular simulation code users implement. Now to prime the model representing corn commodity production simulation, users must add two simple lines of code to help MFDES identify what material flows to track. These lines are Blue in color at the end. The MFDES\_input\_keys variable is to track what are the input material flows required to run the simulation and MFDES\_output\_keys variable is track the material flows coming out the simulation process.

import sys, os

import pandas as pd

import pcse

import numpy as np

import yaml

import datetime

from pcse.fileinput import YAMLCropDataProvider

from pcse.fileinput import CABOFileReader

from pcse.util import WOFOST71SiteDataProvider

from pcse.base import ParameterProvider

from pcse.fileinput import YAMLAgroManagementReader

from pcse.db import NASAPowerWeatherDataProvider

from pcse.models import Wofost71\_WLP\_FD

aux\_info = yaml.load(open('corn\_agro.yaml'))

#corn model

cropdata = YAMLCropDataProvider()

#cropdata

a = cropdata.set\_active\_crop('maize', 'Grain\_maize\_201')

#soildata

soildata = CABOFileReader('ec3.soil')

# weather data

wdp = NASAPowerWeatherDataProvider(latitude=40.4842, longitude=-88.9937)

sitedata = WOFOST71SiteDataProvider(WAV=100, CO2=360)

parameters = ParameterProvider(cropdata=cropdata, soildata=soildata, sitedata=sitedata)

agromanagement = YAMLAgroManagementReader('corn\_agro.yaml')

wofsim = Wofost71\_WLP\_FD(parameters, wdp, agromanagement)

wofsim.run\_till\_terminate()

df\_results = pd.DataFrame(wofsim.get\_output())

df\_results = df\_results.set\_index("day")

#biomass = df\_results.iloc[-1,:]['TAGP']

crop\_yield = df\_results.iloc[-1,:]['TWSO']

#aux data

npk\_applied = aux\_info['AgroManagement'][0][datetime.date(2017, 5, 1)]['TimedEvents'][0]['events\_table'][0][datetime.date(2017, 5, 10)]

N\_applied = npk\_applied['N\_amount']

P\_applied = npk\_applied['P\_amount']

K\_applied = npk\_applied['K\_amount']

MFDES\_output\_keys = [crop\_yield]

MFDES\_input\_keys = [N\_applied, P\_applied, K\_applied]

1. Demonstration of EM Priming with Aspen Plus

The following is demonstrative example of how an Aspen Plus model is primed by using a .csv file as a meta input file. Fig A below shows an Aspen Plus model flow sheet for soybean oil production. In the priming stage, users need to tag individual overall input/output flows with commodity names as shown in table 1 and upload it as a .csv file. Users can also choose to simply use the default flow name of the model if already named based on commodity

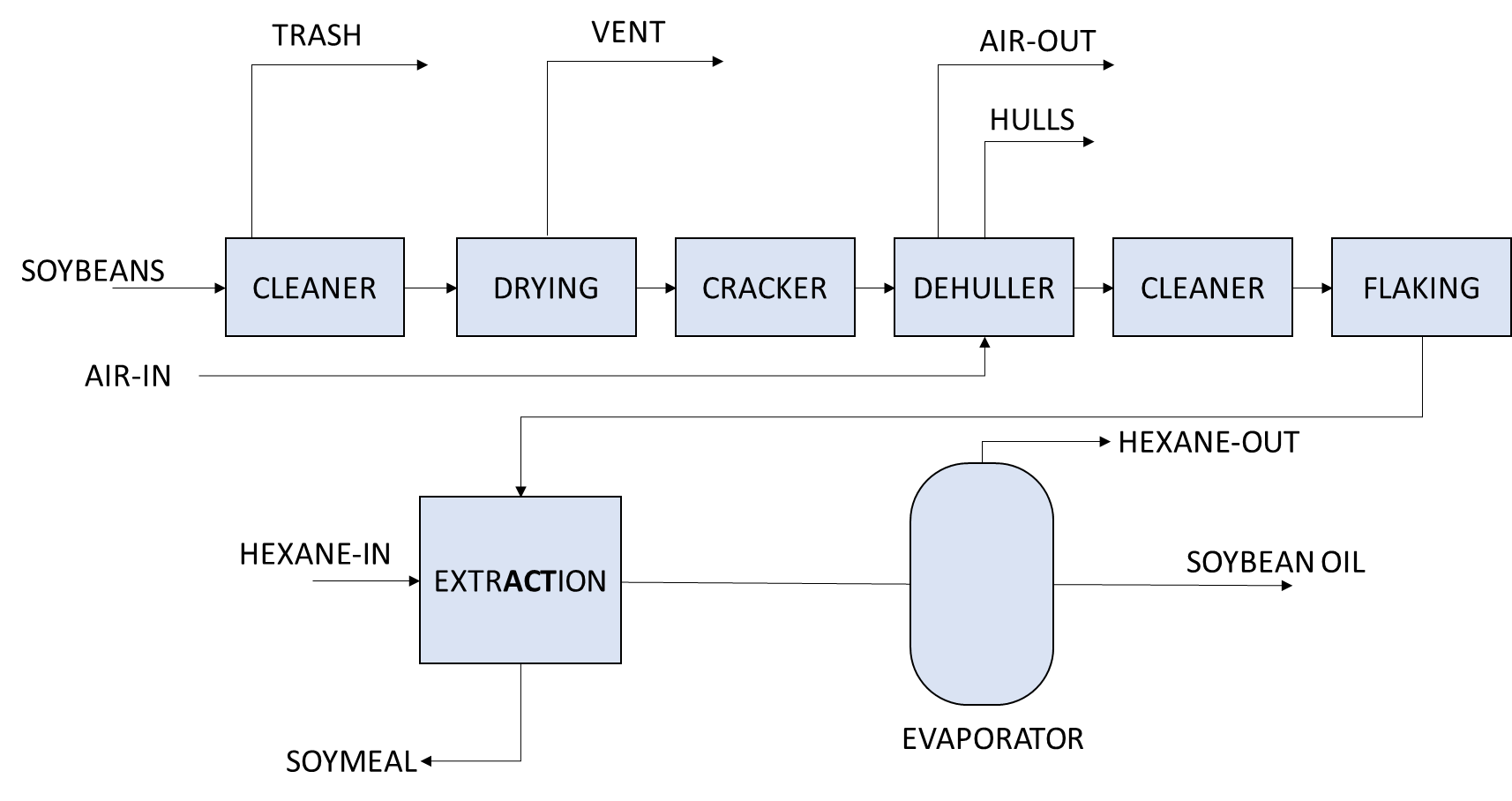


Fig A: Aspen Plus flowsheet for soybean oil production.

|  |  |
| --- | --- |
| **Aspen Plus variable name** | **Commodity name** |
| SOYBEANS | SAME AS MODEL |
| AIR-IN | Air-generic |
| HEXANE-IN | Hexane |
| SOYBEAN OIL | SAME AS MODEL |
| SOYMEAL | SAME AS MODEL |
| TRASH | Waste-generic |
| HEXANE-OUT | Hexane-waste-mix |
| HULLS | Waste-generic |
| AIR-OUT | Air-generic |
| VENT | Waste-generic |

Table 1: Renaming Aspen Plus flows before uploading the model to PIOT Hub tool GUI.

1. Additional information on MFDES tool modules

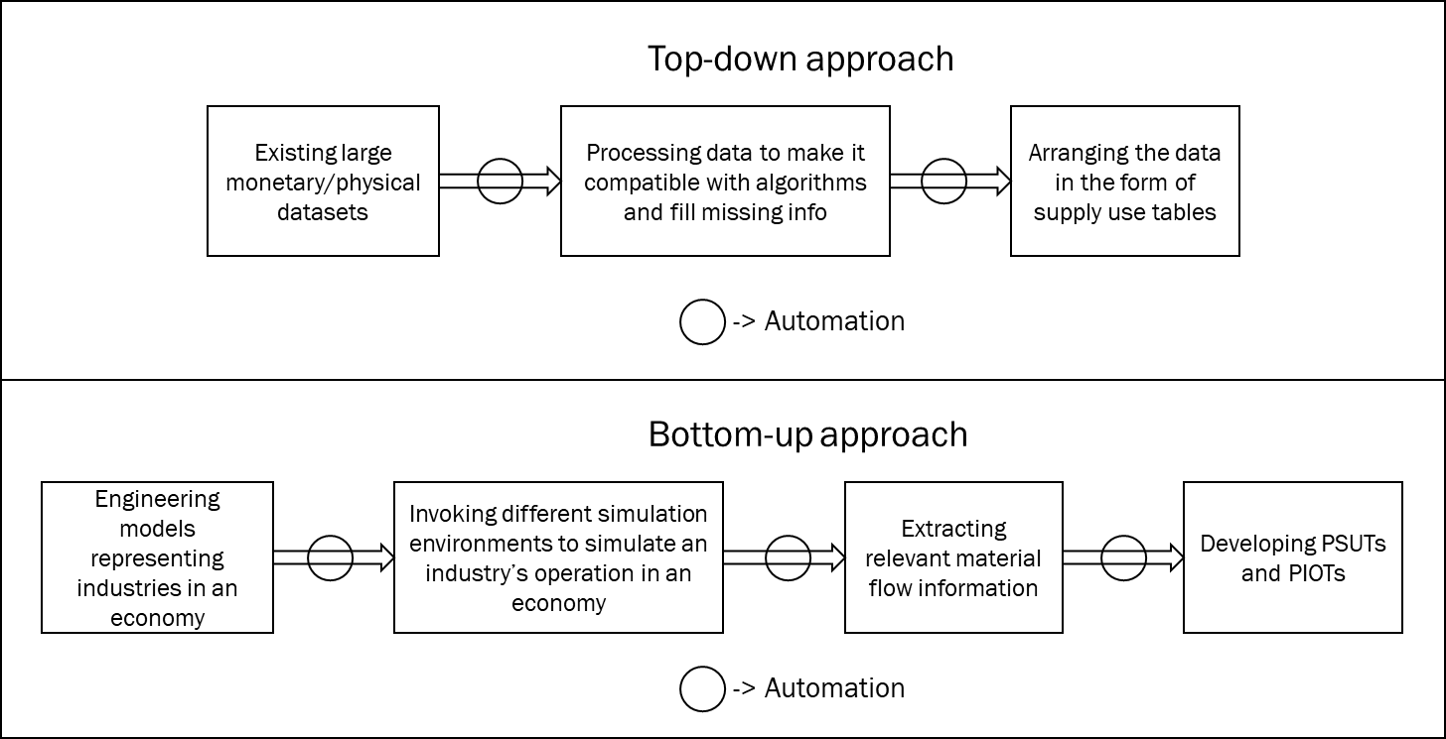


Fig B: The figure above shows the primary differences top down approach and bottom up approach to build PSUTs and PIOTs.

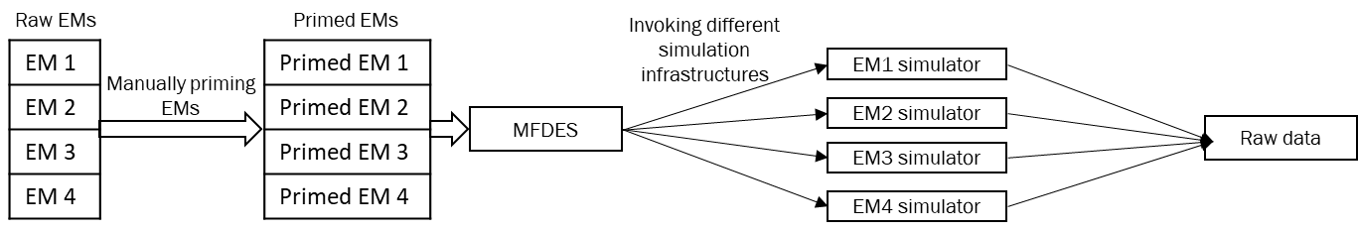


Fig C: The figure above shows a detailed view of various tasks carried out in the simulation module of the MFDES tool. These include identifying the model type, taking a primed model as an input, invoking different simulators based on the model, simulating the models to extract all the output data an storing it in its raw form in temporary memory blocks of a server.

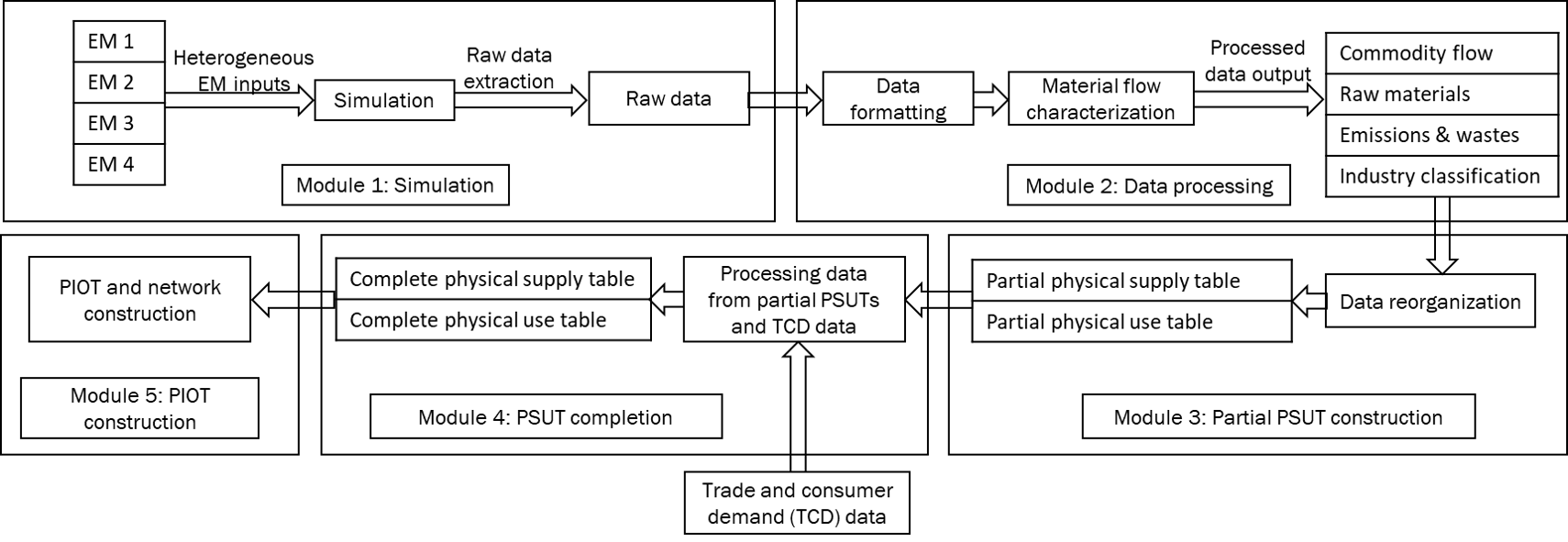


Fig D: Figure above shows all the modules of the MFDES tool without cloud implementation. These are the bare minimum modules the are required to automate the process of PSUT and PIOT generation on a computer. The same tool when implemented on the cloud as PIOT-Hub, it can take in models as input from multiple geographic locations in the world.

1. Figures used in the main manuscript



Figure 1: Overview of PIOT-Hub Infrastructure



Figure 2: PIOTHub: Collaborative cloud implementation of the MFDES tool



Figure 3: Input tab of the PIOTHub



Figure 4: Output tab of the PIOTHub with PIOT view as a table.



Figure 5: PIOTHub output in the form of a time series, network view or heatmap

References

1. Wit, A. (2017). PCSE Documentation Release 5.3, Wageningen university and research.