In [1]:
```
import swolfpy as sp
import swolfpy_inputdata as spid
import swolfpy_processmodels as sppm
import brightway2 as bw2
import pandas as pd
from time import time
import os
import platform
```

In [10]:
```
from IPython.display import Image
pd.set_option('display.max_colwidth',0)
```

In [11]:
```
project_name = 'BaseCase'
technosphere = sp.Technosphere(project_name)
common_data = spid.CommonData()
ProcessMetaData = sppm.ProcessModelsMetaData.ProcessModelsMetaData
```

In [12]:
```
Treatment_processes = {}
Treatment_processes['LF'] = {'input_type':ProcessMetaData['LF']['InputType'] , 'model':sppm.LF()}
Treatment_processes['Composting'] = {'input_tpye':ProcessMetaData['Comp']['InputType'] , 'model':sppm.Comp()}
Treatment_processes['SS_MRF'] = {'input_type':ProcessMetaData['SS_MRF']['InputType'] , 'model':sppm.SS_MRF()}
Treatment_processes['Reprocessing'] = {'input_type':ProcessMetaData['Reproc']['InputType'] , 'model':sppm.Reproc()}
```

In [17]:
```
Processes = ['LF','Composting','SS_MRF','Reprocessing','Collection']
data = sppm.Distance.create_distance_table(process_names=Processes,transport_modes=['Heavy Duty Truck'],default_dist=30)
distance=sppm.Distance(data)
data['Heavy Duty Truck']
```

Out[17]:

	LF	Composting	SS_MRF	Reprocessing	Collection
LF	NaN	30.0	30.0	30.0	30.0
Composting	NaN	NaN	30.0	30.0	30.0
SS_MRF	NaN	NaN	NaN	30.0	30.0
Reprocessing	NaN	NaN	NaN	NaN	30.0
Collection	NaN	NaN	NaN	NaN	NaN

In [18]:
```
Collection_scheme = sppm.SF_Col.scheme()
Collection_scheme[('RWC','SSYW','SSR')]=1
Collection_processes ={}
Collection_processes['Collection'] = {'input_type': [], 'model':sppm.SF_Col('Collection',Collection_scheme,Treatment_processes=Treatment_processes,Dis
```

In [19]:
```
start=time()
```

In [30]:
```
demo=sp.Project(project_name, common_data, Treatment_processes, distance, Collection_processes, technosphere)
```

```
---------------------------------------------------------------
KeyError                  Traceback (most recent call last)
Cell In[30], line 1
----> 1 demo=sp.Project(project_name, common_data, Treatment_processes, distance, Collection_processes, technosphere)

File ~\Documents\Miniconda\envs\swolfpy\lib\site-packages\swolfpy\Project.py:131, in Project.__init__(self, project_name, CommonData, Treatment_process
es, Distance, Collection_processes, Technosphere_obj)
    129 self.waste_treatment = {}
    130 for i in self.CommonData.All_Waste_Pr_Index:
--> 131     self.waste_treatment[i] = self._find_destination(i)
    133 self.process_model = {}
    135 # Creating swolfpy parameter class

File ~\Documents\Miniconda\envs\swolfpy\lib\site-packages\swolfpy\Project.py:153, in Project._find_destination(self, product)
    151 destination = []
    152 for P in self.Treatment_processes:
--> 153     if product in self.Treatment_processes[P]["input_type"]:
    154         destination.append(P)
    155 return destination

KeyError: 'input_type'
```

In [ ]:
```
```