## Concept for integrating the Material Flow Modelling into the GDSE backend

### Requirements and Data Availability

The GDSE so far represents the material flows in the status quo scenario.

For estimating the impact of a solution on material flows (and based upon these modified flows the LCA-Impact indicators), we have to develop a material flow model which is sensitive to changes in input quantities and ratios. Herfore, we could

* import the flows from the database into a representation of a directed graph with nodes and edges in memory
* identify, which nodes are sources (only out-flows) and which nodes are sinks (only in-flows) and which nodes represent activities, that convert in-flows to to out-flows according to certain ratios
* calulate the ratios for each node

This directed graph model then should be able to calculate the flows from the sources to the sinks using the ratios stored for each node.

The next step would be the validation oft he graph model by calculating of the status quo flows and compare the results to the original flow data.

With the validated model, scenario of changed flows will be calculated. The following modifications are foreseen:

* modified source (Double a certain input flow) and compare which flows are affected by the changed input flow
* changed ratio at a node
* enter a new edge between existing nodes and recalculate the ratios on the affected nodes
* enter a new node with new edges and recalculate the ratios on the affected nodes

The results should be then visualized in the frontend and therefore be provided by the backend.

Two general options:

1. Calculate a scenario „on the fly“
2. Precalculate a scenario and store the results of a scenario in the database and query the results of a scenario from the database

The option A) needs a fast calculation oft the scenario. The option B) requires large amount of data to be stored in the database.

The following tasks have been identified so far:

#### select a graph libraty to use in Repair that matches the project requirements

So far, three graph libraries have been identified that could be used to represent the graph model in memory and carry out the material flow modelling:

* Networkx: https://networkx.github.io/
* Igraph: igraph.org/python/
* Graph-tool: <https://graph-tool.skewed.de/>

NetworkX is a pure python project with extensive funcionality. For large graphs it can be slow.

Graph-Tool is written in C++ , the api is avalable in python. Therefore it’s very fast. It can calculate algorithms in parallel using the OpenMP library. There is no precompiled windows version available so far. If we want this (to enable development in a windows environment) we would have to provide a compiled windows version and invest some hours or days in the deployment.

Python-IGraph: written in C, precompiled windows version available

A comparison of the available features hast to be carried out.

#### Convert database objects to graph object

In the Django-Database, the activity groups, activities and actors represent nodes, the flows represent edges of a graph.

The task would be to read the actors and flows from django and convert them into according graph object with vertices, edges and the according graph, vertex, and edge-attributes. The graph-objects can be stored as gml/graphml objects.

One option would be to write a **GML-serializer**, that returns the data in a gml( or graphml/graphmlz-)format that can be consumed by the graph librara. Another option would be to **directly** use the **django models** of the actors and flows and add vertices and edges this way…

#### Analyse graphs

The graphs can be analyzed to identify, which actors are sources and which are sinks (in-degree and out-degree of vertices…)

#### Calculate ratios on nodes

For each edge: calculate a new attributes: ratio\_out and ratio\_in

Algorithm:

For each node:

* Sum\_outflows = sum(all out-flows of outgoing edge)
* Ratio\_out = out\_flow / Sum\_outflows
* Sum\_inflows = sum(all in-flows of ingoing edge)
* Ratio\_in = in\_flow / Sum\_inflows

#### Calculate flows

* Start at origins:
  + Kg of node = amount of origins
* For all outgoing edges of a node:
  + Calculate the kg-attribute of an edge by multiplying the ratio with the kg of the node
* For all other nodes: sum up all kg of incoming edges and set attribute „kg“ of this node

To do this efficiently, search an effective path through the graph, so that a node quickly gets all ist „in-nodes“.

#### API-Call to get results of a scenario (or the status quo)

* Add an additional API-query-parameter to return another scenario for actors, flows etc.
* Write the serializer that returns the scenario results instead of the „baseline data“ from the database.

#### API-Call to create or modify a scenario by a solution

Define, how a solution modifies the flows with a clear syntax

Define, how the solutions are applied in the material flow model…