Basic Input Mode:

User Inputs:

- fusion_power [GW]
- TBR
- reserve_inventory [kg]
- startup_inventory [kg]
- Input_recipe
- fuel_purchase_frequency [timesteps]

Assumed Inputs:

- burn_rate = 55.8 kg/(yr*GW) * fusion_power
- AF = 1 (simplify things to start with)
- τ_{OFC} = 24h (negligible)

Storage:

- Implemented as ResBufMaterialInv()
- Separate storage for tritium, enriched_blanket_material, and _depleted_blanket_material
- Tritium storage has tritium_reserve, tritium-core

Functions:

Breed Tritium(TBR, burn_rate):

- fuel_usage = burn_rate * fusion_power * sec_per_year / dt
- Remove fuel_usage from storage, add (fuel_usage * TBR to storage)
- Tick()

Calculate_output_recipe(input_recipe, TBR, burn_rate):

- Calculates how much Li-6 needs to be converted into T+He-4 to sustain tritium breeding
- Returns output_recipe
- Tick()

Transmute(output_recipe):

- material.Transmute(output_recipe)
- Tick()

Decay():

- material.decay()
- Tick()

Record:

- Feature available through cyclus agent context
- Tick():
    - Datum = self.context.new_datum(“<Name of Table>”)
    - Datum.add_val(“Agent_ID”, self.id, type=ts.int)

Decommission():

- Sell off all tritium in storage and left in the reactor ($I_{Startup}$)

Startup():

- Takes the initial amount of tritium required for startup and “loads” it into the reactor core before the first timestep.

Get_material_requests(self):

- Send out requests for fuel_incommods
- Contains logic for buy strategy
- Returns request portfolio and constraints

Get_material_bids(self, requests):

- Send out offers for fuel_outcommods
- Contains logic for sell strategy
- Returns bid portfolio and constraints

Get_material_trades(self, trades):

- Send out responses for trades
- Returns a list of completed trades (I think?)

Accept_material_trades(self, responses):

- Handles what happens after the trade is finalized (stores materials in inventory of some sort)
- Returns nothing

Notes:

Tritium will be tracked as a “Material” which has a recipe and a mass (in kg). Operations which can be performed on this material are:

- Create
- Extract [Qty/Comp]
- Absorb
- Transmute
- Decay

Lithium will also be treated as a material, which will have a recipe which specifies some enrichment. In the more advanced implementation, the model will transmute a certain percentage of the Li-6 into T + He and lithium enrichment can then be tracked.

Each timestep a certain amount of tritium will burn (in the basic implementation it will be 55.8kg * power/1000MW * 1/12 with a timestep of 1 month or something like that), and then the amount of tritium added to the system will be TBR*amount burned such that the total change in tritium will be:

It may be important to be careful how we do burnup, breeding, and decay sequentially. It is currently unclear if there is a best practice, or if it matters (maybe for really short simulations it does, but for longer ones it's fine?)

Transmute works by changing in_recipe to out_recipe without changing total mass, it requires both inputs.