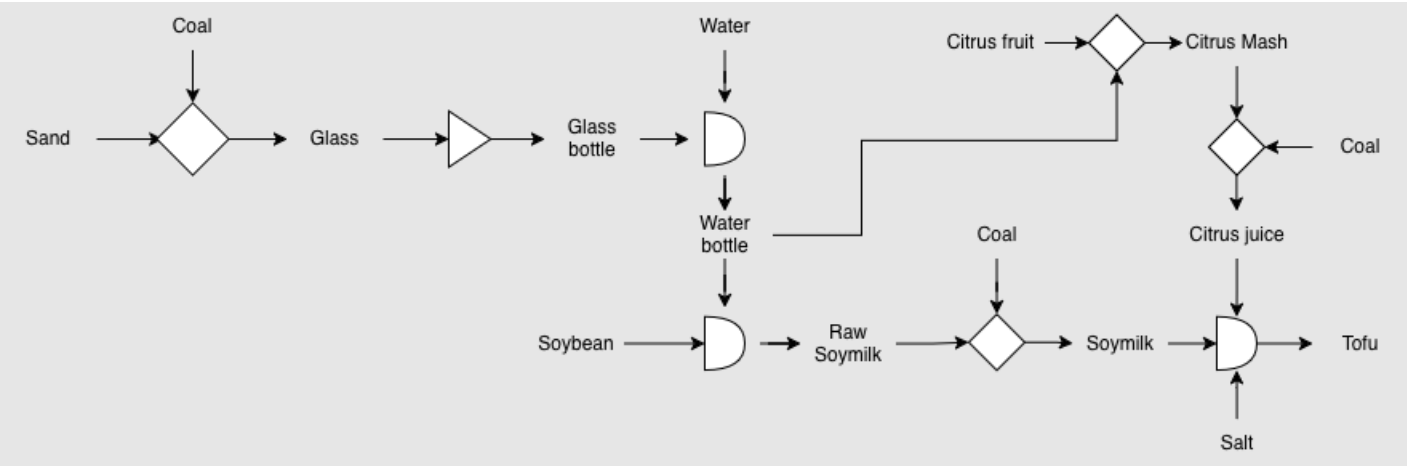
Nalua Valley – goal and main functions

- Value system for farming products
- Register a food item

Value system for farming products

We have to give products made from base items a value, depending on production steps = complexity, energy need.



Formula

Production steps = 1 * nr of steps (≥2)
Energy units = 1 * number of units needed
If we need ≥ 3 base items (e.g. flour, citrus mash), we use the multiplier k. k is for now 1.6 When the input number = output number, no mulitplier (e.g. glass). If we need ≥ 3 different items to make a product, we add j = 0.5 to mulitiplier k

Formula: value = k x production steps x (energy units + 1) / 4
If we don't need any energy, the value equals the production steps * k

As starter: 1 value unit = 1 Minegeld. We can adapt the value then with the divider at the end, for now = 2.

Idea: You can earn money by making high valued products as e.g. soya. Simple products as e.g. flour can not be selled to the bank. Having produced more complex items allows you to create feasts, earning temporary enchantments.

Products

- **Tofu** [11|4] 13.3*5/4 = **17**
  2x {sand >> water bottle [c3|1]}
  {soybean 3 > raw soymilk [1*k|0]}
  {raw soymilk > soymilk [c1|1]}
  {citrus fruit 3 + water bottle > citrus mash [1*k|0]}
  {citrus mash > citrus juice [p1|1]}
  {citrus juice + soymilk 2 + salt > tofu [1*k|0]}
- **Citrus juice** [5|2] = 5.6*3/4 = **4**
  {sand >> water bottle [c3|1]}
  {citrus fruit 3 + water bottle> citrus mash [1*k|0]}
  {citrus mash > citrus juice [p1|1]}
- **Bread** [7.1|3] 7.1*4/4 = **7**
  {Wheat > flour [p1|1]}
  {sand >> water bottle [c3|1]}
  {flour 3 + water + salt > dough [1*k|0]}
  {dough > bread [c1|1]}
- **Polenta in a Bowl** [7.1|2] 7.1*3/4 = 5
  {corn > corn_grains [p1|1]}
  {corn_grains > corn_meal [p1|1]}
  {corn_meal 2 + water bottle + salt + bowl [1*k|0]}

Register a food item

register item as node

Example

```
-- Barley Soup
local barley_soup_def = {
    description = S('Barley Soup – Swiss traditional') .. '\n' .. S('Compost chance') .. ':
100%\n'
        .. core.colorize(xn_farming.colors.brown, S('Hunger') .. ': 6'),
    short_description = S('Barley Soup'),
    drawtype = 'mesh',
    mesh = 'xn_farming_beetroot_soup.obj',
    tiles = { 'xn_farming_barley_soup_mesh.png' },
    use_texture_alpha = 'clip',
    inventory_image = 'xn_farming_barley_soup.png',
    wield_image = 'xn_farming_barley_soup.png',
    paramtype = 'light',
    paramtype2 = 'facedir',
    is_ground_content = false,
    walkable = true,
    selection_box = {
        type = 'fixed',
        fixed = { -0.5, -0.5, -0.5, 0.5, 0.1, 0.5 }
    },
    collision_box = {
        type = 'fixed',
        fixed = { -0.5, -0.5, -0.5, 0.5, -0.1, 0.5 }
    },
    groups = {
        -- MTG
        vessel = 1,
        dig_immediate = 3,
        attached_node = 1,
        -- X Farming
        compost = 100,
```

```
        -- MCL
        food = 3,
        eatable = 6,
        compostability = 100,
        handy = 1,
        deco_block = 1,
        fire_encouragement = 60,
        fire_flammability = 100,
        dig_by_water = 1,
        destroy_by_lava_flow = 1,
    },
    sounds = xn_farming.node_sound_wood_defaults(),
    sunlight_propagates = true,
    on_use = core.item_eat(6, 'mcl_core:bowl'),
    -- MCL
    _mcl_saturation = 0.6,
    _mcl_blast_resistance = 0,
    _mcl_hardness = 0,
}
if core.get_modpath('mcl_farming') then
    barley_soup_def.on_secondary_use = core.item_eat(6, 'mcl_core:bowl')
end

core.register_node('xn_farming:barley_soup', barley_soup_def)
```

Register craft for this item

```
-- barley soup, a typical swiss recipe :-)
core.register_craft({
    output = 'xn_farming:barley_soup',
    recipe = {
        { 'mcl_farming:carrot_item', 'xn_farming:salt',
'xn_farming:barley','mcl_potions:water','mcl_core:bowl' },
    }
})
```