Oikosol

Oikosol Design Guidelines

23rd November 2014

The following document intends to describe guidelines and good design practices for designing Oikosol technology (1), as well as the instructions (2) describing its construction. These guidelines are intended as a recommendation rather than a limitation, and this document might undergo changes in the future to accommodate for new ideas.

Three Main Objectives

Oikosol technology and its instructions have three main objectives:

- Accessibility (to as many people on Earth as possible);
- Ease of assembly and ease of use;
- Durability.

1) Technology Guidelines

a) Design for the world: simple materials, easy to get or create from raw local materials;

b) Aesthetics are important inasmuch as they don't restrict guideline a);

c) The end goal of each technology is to *produce* as much of the desired outcome (power, clean water, food, treated waste and transport potential) as possible;

d) The technology should be as durable as possible. We reject planned obsolescence;

e) The technology should be easily transportable whenever possible and whenever this does not conflict with guideline c);

f) The technology components should enable the use of other open-source components as much as possible, as well as the possibility to 3D print most of the components.

2) Instruction Guidelines

a) Create the instructions in an accessible format, such as a free word processor compatible with most common word processors;

b) Include clear designs of the final assembled technology and all the steps in between;

c) Replace written instructions with figures and steps that don't require language. Whenever this cannot be performed, use the English language;

d) If you are editing an existing instruction, don't forget to record the most relevant changes you have made from the previous version;

e) Always use either the Metric System or the International System of Units when describing components/assembling in their fourth dimensions (height, length, depth and time). For example, using centimeters, millimeters, meters, seconds, minutes, days and weeks.

f) If you are creating a new technology or a different approach/design to an existing technology (example: vertical wind turbine vs horizontal wind turbine, Media based aquaponics vs DWC based aquaponics), you can number your version as v.1. If you are building upon previous work, follow the general rules of semantic versioning of software:

> f.1) When you introduce a MAJOR change (which makes some previous base components obsolete or requires new base ones), then introduce the next number. If the previous version was v.1 then your version will be v.2 and so on;
>
> f.2) When you introduce a MINOR change (which adds functionality in a backwards-compatible manner, through re-arranging the design or adding/removing minor new components), then introduce the next number after one decimal point. If the previous version was v.1 then your version will be v.1.1, then v.1.2, v.1.3 and so on. It is assumed that v.1 is the same as v.1.0 in this context;
>
> f.3) When you introduce a PATCH change (which is intended to solve an issue of the previous design, as a backwards-compatible bug fix), then introduce the next number after two decimal points. If the previous version was v.1.1 then the patch will be v.1.1.1, if the previous version was v.3.2.2 then the patch would be v.3.2.3 and so on. It is assumed that v.1.1 is the same as v.1.1.0 in this context.

g) Future proposed template changes to the Soleco v.1 instruction template should have in mind our three main objectives at all time.