Write-up for Top-down Iterative Refinement (TDIR)

**General Considerations:**

Top-down iterative refinement focus on importance leveling and largest conceptual module decomposition.

Top-down iterative refinement technique is mainly focus on group modules by their importance. Which means the importance of each module will gradually decreasing from left to right; the modules with same importance will be group on same level. Top-down iterative refinement is a good practice when the design is more hierarchical, this design technique will show a partial factored structure and also define the largest and most complex module in the among the system. During the design using TDIR, designer needs to decompose the largest chunks (module) and define it as next level, so it is easy to identify whether the Miller’s Law is violated.

**Analysis Metrics:**

1. *Coupling*

Coupling is a measure of inter modular dependence. It may be because of shared content, shared data, or control. As a design principle, it is desirable to minimize coupling.

1. *Miller’s Law*

Miller’s law considers the number of components or functions within each module is around 7 ± 2. To accomplish this, designer needs to clearly know the function of each module, and control the functionality of each module within 7±2.

1. *Gracinuas’s Law*

Gracinuas’s Law consider that interrelation between modules. By using formula to find the number of interactions that may influence the design. The best way to show the interrelation is to mark the data flow and the control flow between modules.

1. *Factoring*

While factoring a design, the upper level modules tends to have more control and lower level modules tends to have more implementation detail. System that are highly factored will have more detailed module than the control module. A well factored system will also have significate effect on the afferent input and efferent output.

1. *Scope of effect*

Scope of effect of a module is defined as all of the modules affected by a decision made by the module, and scope of control of a module is defined as the module itself and all of its subordinates. The design goal is let scope of effect to be subset of the scope of control. In order to accomplish this, the design must to be hierarchical with control flow.

1. *Black Boxes*

The Rule of black boxes is to focus on the design of the system without concern the structural or methodological realization. A true black box can be fully utilized without knowledge how it is constructed. It exhibits high integrity and does exactly same function for every module that invokes it.

1. *Fan-In/Out*

Fan-in is primary goal of design. This means the result and data merged into one module which will decrease the cost of maintenance and modification in long-run. The less desirable situation is if fan-in and fan-out are both target same module, that means there is multiple idea in one module and it needs to be test by Graicunas’s law.

**Analysis:**

1. *Coupling*

Since Top-down iterative refinement will not show any data flow or control flow so there is no detection on the data and control coupling.

1. *Miller’s Law*

Since the top-down iterative refinement will identify the largest conceptual module in each level, so it is very clear to show the amount of functions within each modules. For our project, the largest module is engine, and the engine will contain around 5 different functions. Thus, our project is not subject to the effects of Miller’s law.

1. *Gracinuas Law*

Because top-down iterative refinement will not specify any data flow and control flow so it is failed to test by Gracinuas law. Sine there is no relationship to show, we cannot use Graicunas Law.

1. *Factoring*

TDIR is a good practice for the factoring. The reason is TDIR will decompose the largest conceptual component and define it as next level of the structure. This will construct a partial well-factored system. As the importance of the system is gradually decreasing from left to right, we could have a general structure of the design. However, it is not a good practice when the design needs more specific structure and the necessity to show the data flow and control flow. For our project, we need to specify the clear structure with data flow and control flow. So the TDIR could be a guide line during the design phase.

1. *Scope of control/effect*

There are three identities that must be applied if we want to use the scope of control technique. The system must be hierarchical; Must define the decision module and the modules that affect by the decision; also the data and the control flow. Since TDIR will not show any of the data and control flow in the diagram and there is no direct indication of scope of control and the scope of effect, so TDIR cannot be measure by scope of control/effect.

1. *Black Boxes*

TDIR is a good practice if measured by the black boxes technique. Since TDIR only focuses on the decomposition of the largest conceptual module and leveling them by their importance. Therefore, the implementation is irrelevant to TDIR. From that, it remains the integrity and the confidential of each module. For our project, we start our design by using the TDIR because it allows our design team focus on the design rather than the implementation. Thus, TDIR is a valid way to apply the black boxes principle.

1. *Fan-In/Out*

Fan-in and fan-out require to show a hierarchical structure with data flow. TDIR is not a good practice for our project because it is hard to show the fan-in and fan-out of system due to only the largest module will be decompose to next level. Since top-down iterative refinement will not decompose smaller conceptual branches, and there is no indication of data flow, so TDIR cannot be measured by fan-in/out structure.

**Conclusion:**

From analysis we know: TDIR is not a good practice when the design is more emphasis on showing the interrelation, such as I/O, data and control flow. So there is no way to show whether the data flow or control flow have coupling. TDIR is not a good choice when the design that tends to have more control and data flow, as well as the design that needs more flexibility rather than integrity. TDRI have the advantage to show parent-child relationship between each module, but since game engine tends to have more control and data flow, so the TDRI is not a good technique for our team project.