**Reviewer(s): Jason Stidham, Lansdon Page, Ryan Wilson, Jason Wong Date(s): 9/28/2014**

Inspection Checklist for Software Design Specifications (SDS)

Reviews for software design focus on data design, architectural design, and procedural design. In general, two types of design reviews are conducted. The preliminary design review assesses the translation of requirements to design of data and architecture. The second review, often called design walkthrough, concentrates on the procedural correctness of algorithms as they are implemented within program modules. Below you can find two checklists, one for each kind of design review.

## (1) Preliminary Design Review

**Traceability**

Did you ensure traceability of the design back to the systems specification and statement of requirements? See RTM.

* All specifications identified in document satisfy a requirement listed in the Requirements specification document.

Is a scheme used for naming of modules, data, and interfaces?

* All modules, attributes, and methods follow a consistent template throughout the document.

~~Are all modules, data, and interfaces uniquely identified?~~

* Not applicable to product. Many modules deliberately utilize duplicate names for their methods and attributes because it improved readability of the program as well as consistency with the behavior of the modules.

**Consistency**

Is the data structure consistent with the information domain?

Is the data structure consistent with software requirements?

Is a standard design representation used?

Is a standard data usage representation used?

**Completeness**

Are software requirements reflected in the software architecture?

Are all referenced data defined?

Are all defined data used?

Are all referenced modules defined?

Are all defined modules used?

Are interfaces defined for modules and external elements?

Has maintainability been considered?

Have quality factors been explicitly assessed?

**Efficiency**

Are data grouped for efficient processing?

~~Are storage requirements allocated to design?~~

Is effective modularity achieved? Are modules functionally independent?

**(2) Design Walkthrough**

Does the algorithm accomplish the desired function?

* The product contains the necessary modules to imitate the behavior of a Universal Turing Machine.

Is the algorithm logically correct?

* The product contains the necessary elements required for a Universal Turing Machine.

Is the interface consistent with the architectural design?

Is the logical complexity reasonable?

* Product is loosely coupled, and has both its conceptual (Requirements) and implementation (Design) behavior documented.

Has error handling been specified?

* Error handling isn’t explicit. There are details explaining how certain elements of the product “should” behave, but none in regards to how it does.

Are local data structures properly defined?

Are structured programming constructs used throughout?

Is design detail amenable to implementation language?

~~Has maintainability been considered?~~

* Product was not created with the notion of being used beyond the month of May 2014.

Are all conditions and processing defined for each decision point?

Do all defined and referenced calling sequence parameters agree?

Appendix A – Requirements Traceability Matrix (RTM)

The following table is an example of what may be utilized for a Requirements Traceability Matrix (RTM). This may be customized to fit the stakeholder’s needs.

| ID | System Requirement | Use Cases | Design Elements | Test Cases |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
|  |  |  |  |  |