Software Architecture & Design (SAD) Document v.1.X

Project Name: Date:

Project Manager:

Assistant Project Manager:

Software Engineer #1:

Software Engineer #2:

…

Software Engineer #n:

1. Purpose – What’s the purpose of this document in the development of the software and what’s the purpose of this document in maintenance.
2. High-Level Design – Usually this is in the form of UML component diagrams or package diagrams, which shows the major modules of the software being developed and the interfaces (ball and socket) for component diagrams or dependencies (dotted lines with arrows at the end) for package diagrams. There are several architecture styles you can use, such as hierarchical decomposition (or top down), layered architectures (3-tier or 4 tier), client/server architecture, repository architecture, and Model-View-Controller architecture. The high-level design begins from the deployment diagram that’s defined in the SRS. Inside each hardware depicted in the deployment diagram is a component that contains the software that is running inside the hardware. Either you use hierarchical decomposition to determine the details of each component or use an architectural style that fits the need of the software design.
3. Low-Level Design – Usually this is in the form of UML class diagrams showing the outline of a class, such as classname, visibility, list of member variables, and list of member functions. The class diagram shows the different classes inside a component diagram or package diagram that were defined in the high-level design. We do this **for each** component or package.
4. Pseudo-Code Algorithms of Critical Functions in Class Diagram – **For each** class depicted in the low-level design, we write the critical/important functions in pseudo-code, which depicts the logic underlying the correct operation of the function.