**1. INTRODUCTION**

**1.1 PURPOSE**

The goal of these guidelines is to create uniform coding habits among team so that reading, checking, and maintaining code written by different persons becomes easier. The intent of these standards is to define a natural style and consistency. When a project adheres to common standards many good things happen: Programmers can go into any code and figure out what’s going on, so maintainability, readability, and reusability are increased. Code walk trough’s become less painful. New people can get up to speed quickly. People new to a language are spared the need to develop a personal style and defend it to death. People new to a language are spared making the same mistakes over and over again, so reliability is increased. People make fewer mistakes in consistent environments. Idiosyncratic styles and college-learned behaviours are replaced with an emphasis on business concerns - high productivity, maintainability, shared authorship, etc. Experience over many projects points to the conclusion that coding standards help the project to run smoothly. They aren’t necessary for success, but they help.

A mixed coding style is harder to maintain than a bad coding style. So it’s important to apply a consistent coding style across a project.

Since a very large portion of project scope is after-delivery maintenance or enhancement, coding standards reduce the cost of a project by easing the learning or re-learning task when code needs to be addressed by people other than the author, or by the author after a long absence. Coding standards help ensure that the author need not be present for the maintenance and enhancement phase.

**2. FILE and MODULE GUIDELINES**

**2.1 Module Design Guidelines**

All source code will be grouped into modules. Each module will deal with a single, unique domain.