Systems Dev – Coding Issues

- Lots of code do not have unit test (less than 30% code coverage)
- Some code do not have API doc/Javadoc/PHPdoc/Doxygen
- Lots of functions/methods/routines do not have remarks/comments
- English is not the reference language to code
- Non-standard coding styles (Use Coding Standards and Conventions**)**
- No code profiling (dynamic code analysis)
- No nightly build
- Repetition of code (use DRY not WET)
- Not using software design pattern

Systems Dev – No STUPID code

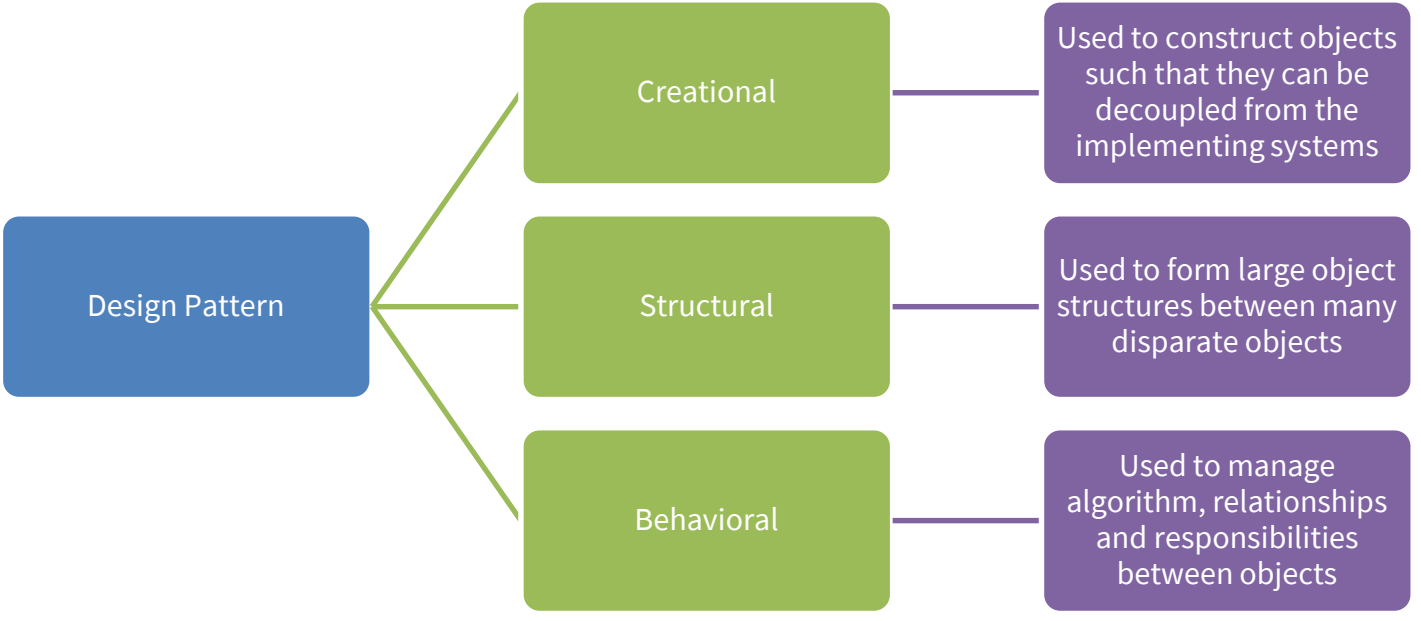
Avoid writing STUPID codes.

What's a stupid codebase? A STUPID codebase is one that uses:

Singletons - Most times we need more than one instance of a class.
Tight coupling - Why link two components in such a way that changing one will not work unless the other is changed?
Untestable code - Shipping without writing tests is a recipe for broken deployments.
Premature Optimisation - Don't waste your time trying to optimize parts of the codes that do not improve the overall software performance.
Indescriptive Naming - Naming can be a pain sometimes.
However, it is more painful to give names that do not describe well what the class, function, method,
Duplication - Copy and paste may seem easy but truly it makes the code ugly.
You have a long codebase full of redundant codes. Learn to make it DRY

Systems Dev – SOLID Principle in Programming

Systems Dev – Software Design Pattern



Version Guide

major.minor.maintenance/revision.build
example: **1.2.1.0** (Alpha)

Build Status for:
0 for alpha -> for development/SIT
1 for beta -> for UAT
2 for release candidate -> for Prod Deployment
3 for (final) release -> Stable release

- Put in footer application
- Tags in code repositories