**8. Identify and document how these principles help to avoid code smells.**

**a. SRP**

**b. OCP**

**c. ISP**

**Three Simple Ways to Avoid Making Code Smells**

I was pondering about how quick hacks can contribute to code destabilization and the previous

Post The Dangers of Git R’ Done. How many hacks or code smells should be allowed per project?

The obvious theoretical answer is none, but the realise answer is too many. Hare are 3 easy ways to reduce the number of hacks and code smells in your projects.

**1.Eliminate gold plating**

I looked over the artifacts for my last couple projects in order to find some insight. Habitually, I allowed 1 – 2 moderate to high smell hacks per project. Further investigation found these hacks were implemented during the last iterations of each respective project.

Late game hacks are not an uncommon thing; however I did find the correlation between end game gold plating and these hacks interesting. Over the past couple of years, every hack was associated with a gold plated requirement.

The mere speak of the acronyms yagni and kiss should send gold plating packing. Awareness of budget and quality impacts is also great motivation for allowing gold plating and feature creep.

**2.Negotiate for time to refactor**

Lots to do, with very little time. How you handle this situation will determine if you will be adding new feature or new code smells.

Negotiate with the client or management to prioritize the remaining requirements. This will give you the time you need to complete as many as you can correctly.

**3.Go heavy on unit testing and documentation**

Push comes to shove and it is time to throw in a hack. Unit test it harder than normal, and never assume people will be able to follow the hack through code so document it more than normal. Personally, I explicitly use the notation //hack just as much as I use //TODO.

In short, code smells and hacks are just like mother-in -laws. The best way to handle them is to just avoid them all together. If that is not possible, negotiation, documentation, and testing will make a ton of difference. Happy Coding.