Data Abstraction

Let us consider the following scenario: We have a code whose purpose we wish to hide from the client/user. For example, we wish to hide what resource is used by skill “A”.

For example, the simple following pseudo-code:

Assume we wish to hide the fact this skill uses mana. One method of doing so is to replace the name by another word which only we know, for instance, “Bus”. So, we get the following pseudo-code:

The above abstraction method may seem like a good solution; however, it has an inherit problem: Assume developer and developer work on the following segments of code:

|  |  |
| --- | --- |
|  |  |
|  |  |

Also assume both developers used the same abstraction “Bus” resulting in the following codes:

|  |  |
| --- | --- |
|  |  |
|  |  |

Now, developer gets the code, but does not know to whom the code belongs. In that case, developer cannot interpretate correctly this code since “Bus” can be interpretated to “Health” or “Mana”. Therefore, such an abstraction method has an inherited limit, not to mention that if the interpretation of “Bus” is lost somewhere, then there is no way to recover the code meaning anymore.

My method

Now I present the result of my method, without showing how it is created (shall you be interested to see how to create it, contact me please).

Using this method, we can both abstract and encrypt the words by using a one-to-one function which converts each word to a number.

For example, we use the codes described previously and abstract the data “Mana” and “Health”, resulting in the following encrypted and abstracted data:

|  |  |
| --- | --- |
|  |  |
|  |  |

Where represents “Integer” (since in most coding languages a variable must start with a letter).

As follows, using the decryption method and reverse abstraction, we will always get the result back “Mana” and “Health”.