screen (although there is almost zero delay in the message being received by the recipient). When a message is received, it is first added to the chat box so it is displayed on the screen. It is then sent to a message receiver handler, which as with sending calls the obfuscator class for deobfuscation. This time the message is not sent, and all that is returned is the bits that have been recovered. These are sent back down to the user and displayed in the recovered bits box in the chat form.
If there are no bits in the bitstream then the algorithm by default attempts to hide a 0 in every word when obfuscating. This is due to the fact that the deobfuscation method expects data to be contained in every word, and so if there was no data it would be a lot less predictable.

4.4 The Algorithm

4.4.1 Algorithm Operation

The algorithm operates within a number of standard Java Classes. There are two main classes, one for processing the text and generating the synsets (the WordQuery class), and one for performing the actual obfuscation (the Obfuscator class).
The WordQuery class contains a number of methods for processing the text. The primary method, doQuery(), takes the cover-text, bitstream and the operation to be performed as parameters and then converts the text and bitstream into two separate ArrayList objects. The method then either performs obfuscation or deobfuscation depending on the operation parameter. The synset is then generated as described in sectiopn 5.4.2 below.
The list of returned FrequencyString objects is passed to the Obfuscator class. This class simply returns the element in the list at position 0 for bit 0, at 1 for bit 1, and at 2 for the bits 1,0. The deobfuscation method also in this class, also takes the in-putted word as a parameter and returns its position in the list.
The order of operation in obfuscation and deobfuscation is slightly different. To obfuscate, the synset is generated, and then the obfuscation method called. The ability to deobfuscate is then tested by generating the synset for the result and passing it to the deobfuscator method in the Obfuscator class, and if the bit that is retrieved is the same as the one that was hidden then the test has been passed. If this test is passed, then the quality test is performed using the code below:

```
int quality = 0;
if (i == 0 && input.length > 1) {
        quality = qualityTest(result, input[1]);
} else if (i > 0 && i < input.length - 1) {
        if (input[i - 1].trim().endsWith(".")) {
                if (i < input.length - 2) {
                                quality = qualityTest(result,
```