**AND and OR For Humans and Computers, Efficient Use of AND and OR Decisions, and True Statements are Better than NOT Statements**

David LaCivita

CTIM 281 Software Design and Development

Professor Chuck Bisson

October 6, 2020

**AND and OR For Humans and Computers**

English speaking humans can easily understand the incorrect use of “and” and “or” but computers are not able to. We recently opened a new bank account that offered “no ATM fees” though there were a few requirements. As the employee pointed to the bullet points in their brochure she told us we would get all of our fees refunded at the end of the month if we “...maintained a balance over $1000 and if we made more than fifteen ATM transactions a month and if we used only their ATMs we wouldn’t receive any fees at all.” Of course, we understood this but bankers are not programmers. Luckily, programmers are humans.

For a programmer, the first step is to fully understand the problem. In this case the programmer needs to know what conditions qualify a customer for a refund of ATM fees. Instead of using the info given by the employee the programmer could check written policy or speak to a higher level employee. Once the programmer confirms the conditions needed to receive a refund the programming can begin

**Efficient Use of AND and OR Decisions**

If the bank confirms a customer needs to have a balance over $1000 or more than 15 ATM transactions to receive a refund then we know where to begin. To efficiently write the code for the ATM fee refund program it helps to have some data. Customers who only use the local ATMs receive no fees so we don’t need to worry about refunding them. The bank tells us that 80% of their customers have more than 15 ATM transactions but only 40% keep their balance over $1000. Since we learned this is an OR decision we should start with the number of customers who have more than 15 ATM transactions. For 1000 customers we know 800 of them will receive a refund for having over 15 ATM transactions in a month. This leaves only 200 customers passed on to the next decision. If we asked about the $1000 balance first we would be passing 600 customers on to the next decision. The pseudocode should look like this:

If transactions > 15 then

Issue refund

Else

If balance > $1000 then

Issue refund

End if

End if

If we had learned a customer needs to have a balance over $1000 and more than 15 ATM transactions then things would look a little different. With the same data set would not want to begin with customers who have more than 15 ATM transactions. That would send 800 customers to the next decision. Instead, we should begin with the $1000 balance. Using the $1000 balance we only send 400 customers on to the next decision. The pseudocode should look like this:

If balance > 1000 then

If transaction > 15 then

Issue refund

End if

End if

When the situation requires an OR decision it is more efficient to begin with the more likely response. If it requires an AND statement then we should start with the less likely response

**True Statements are Better than NOT Statements**

Using NOT statements makes logic difficult to read and follow. It may also lead to confusing double negatives which can lead to logic errors. A simple example would be: charge admission to any customer who is not over 65 years old. It would be easier to say customers over 65 get in free. Photography is complicated enough so we should avoid saying: if your shutter speed is not under 1/60 of a second you do not need to raise your ISO. It is much easier to understand if you say: if your shutter speed is over 1/60 of a second then don’t raise your ISO.

Using AND and OR statements in programming should be clear and make it easy to follow the flow of logic. NOT statements can lead to double negatives and make it harder to follow the logic. Often, the thing that complicates the logic the most is taking verbal instructions from a human. It is essential to have as much data as possible and a programmer should take the time to fully understand the problem to make efficient use of AND, OR, and NOT statements.