Code smells : Class activity

```
/// <span class="code-SummaryComment"><summary></span>
/// Validate UI Request and throw a business exception if there is any error
/// <span class="code-SummaryComment"></summary></span>
/// <span class="code-SummaryComment"><param name="request">The business process request.
</param></span>
/// <span class="code-SummaryComment"><param name="customerIDFieldLength">Length of the
Customer field.</param></span>
/// <span class="code-SummaryComment"><param name="productFieldLength">Length of the product
field.</param></span>
/// <span class="code-SummaryComment"><returns>Returns true if the Request was validated, else
false.</returns></span>
private bool ValidateRequest( CustomerInquiryRequest request,
        int customerIDFieldLength, int productFieldLength )
{
    if( request.CustomerProduct.ProductNumber != null &&
        request.Customer.CustomerID != null )
    {
        if( request.CustomerProduct.ProductNumber != string.Empty &&
            request.Customer.CustomerID != string.Empty ){
            // Both were populated
            throw new BusinessException(
                  HandledErrors.InvalidBothParameterMessage );
        }
        else if( request.Customer.CustomerID == string.Empty &&
            request.CustomerProduct.ProductNumber == string.Empty ) {
            // if objects were instantiated but not populated
            throw new BusinessException(
                  HandledErrors.CustomerEmptyMessage );
        }
        else if( request.Customer.CustomerID != string.Empty ){
            // Note: ProductNumber was equal
            // to string.empty Check Customer ID length
            if( request.Customer.CustomerID.Length >
                            customerIDFieldLength ){
                throw new BusinessException(
                      HandledErrors.CustomerInvalidLengthMessage );
                }
        }else{
            // Note: CustomerID was equal
            // to string.empty check Product Length
            if( request.CustomerProduct.ProductNumber.Length >
                                        productFieldLength ){
                throw new BusinessException(
                      HandledErrors.ProductInvalidLengthMessage );
```

Code smells : Class activity

```
            }
        }
    }else if( request.CustomerProduct.ProductNumber == null &&
        request.Customer.CustomerID == null ){
        // Both were null
        throw new BusinessException( HandledErrors.CustomerEmptyMessage );
    }else if( request.CustomerProduct.ProductNumber == null ){
        // ProductNumber was null and CustomerID was not null
        if( request.Customer.CustomerID.Length >
                        customerIDFieldLength ){
            throw new BusinessException(
                  HandledErrors.CustomerInvalidLengthMessage );
        }
    }else{ // ProductNumber not null and CustomerID was null
        // Check Product Length
        if( request.CustomerProduct.ProductNumber.Length >
                                    productFieldLength ) {
            throw new BusinessException(
                  HandledErrors.ProductInvalidLengthMessage );
        }
    }
    // Set objects below with formatted data i.e PadLeft
    if( request.Customer.CustomerID != null ){
        request.Customer.CustomerID = request.Customer.CustomerID.PadLeft(
            customerIDFieldLength,
            Convert.ToChar( "0", CultureInfo.CurrentCulture ) );
    }
    if( request.CustomerProduct.ProductNumber != null ){
        request.CustomerProduct.ProductNumber =
            request.CustomerProduct.ProductNumber.PadLeft(
            productFieldLength,
            Convert.ToChar( "0", CultureInfo.CurrentCulture ) );
    }
    return true;
}
```

1- How easy it is to understand and figure out what the method is supposed to do.
2- Identify common design problems (Code smells) in this object-oriented code.
3- Think about solutions to fix some code smells