Lambda) based upon our actual use. If you make the code more efficient then you pay less. There is suddenly a financial reason for refactoring code. There are many other benefits with such platforms from consuming services to code re-use but the changes to the way we write, refactor and monitor code are significant. This is what co-evolution is all about and in this case, it's the collision between development and finance.

The second thing to note is that marketing is a net loss. How is that possible when in the in-house variant it's positive? On a consumption basis, the cost to acquire and cost of operation for each new user significantly exceeds the additional revenue they create and so it's a loss at this acquisition price. The marketing proposal doesn't make sense in the public platform variant because there's direct linkage of actual cost against revenue. But in the in-house variant, then most of the costs of operation have already been spent in the initial upfront investment. It's a sunk cost. In which case given we've already spent most of the money and we're actually comparing the acquisition cost versus the additional revenue. The marketing proposal makes sense in the in-house variant precisely because you've already blown most of the cost.

But hang on, the third option of both marketing and development looks better than all of them. How can that be? In this case, the reduced cost of each user on the service (because of refactoring i.e. the development effort) means that the total cost per new user (i.e. marketing acquisition plus operational) is now less than the additional revenue they create. The sum of the whole is greater than the sum of