Vehicle Identifier solution document

**Problem**:

Please write a program that can identify vehicle types by their components. Vehicle types are identified by their properties, such as frame material, number and material of the wheels, and power train. Given an input containing this information the program should be able to say what the vehicle type is (Big Wheel, Bicycle, Car, ...).

**Solution / approach:**

As of now, the vehicle has 3 parts on which we need to calculate its type. Thus the static / mapping data about the same will be in say DB. We will read that data and store it in some static map(s) in format like vehicleAttribute VS vehicle type as bellow.

Thus we will have three maps like frameVsVehicleTypeMap, PowerTrainVsVehicleTypeMap, wheelsVsVehicleTypeMap.

Now, based on input vehicle(s) we need to fetch and get the list from each search parameter and then finally intersect all the lists so that we can get the vehicle who matches all the search criteria.

E.g.:

We have vehicle having 2 wheels, power train human and frame as metal.

Then we will fetch vehicles for **2\_wheels** and we will get **motorcycle and bicycle**. After that we will also fetch vehicles for **human** powertrain and we will get **big wheel and bicycle**. Then we will fetch vehicles for frame as **metal** and we will get **car, motorcycle and bicycle**.

**Design considerations:**

In future suppose, more number of the input search criteria is added for vehicle (i.e. vehicle xml has more tags for each search parameter) say, engine, fuel, etc. then in that case searching the vehicle based on all those parameters will be slow / complex and time consuming.

This approach uses reverse index. The maps above which we are already maintaining will be maintained for new attributes as well. Thus at run time i.e. at evaluation time, we just need to get the data from map(s) and intersect it so that it is really fast in term of operation / time consumption. It is scalable as we can maintain the mapping for each attribute we will add in future.

**Issue with approach:**

Having multiple maps in java will consume lot of memory and can cause memory issues.

**Solution to the issue:**

We can maintain the same map / mapping in EhCache and update / refresh EhCache after certain interval periodically or by scheduling some job.