carbon_model.rb

```ruby
module BrighterPlanet
  module Motorcycle
    module CarbonModel
      def self.included(base)
        base.decide :emission, :with => :characteristics do
          committee :emission do # kg CO2
            quorum 'from fuel', :needs => [:fuel_consumed, :emission_factor] do |characteristics|
              characteristics[:fuel_consumed] * characteristics[:emission_factor]
            end
          end

          committee :fuel_consumed do # litres fuel
            quorum 'from distance and fuel efficiency', :needs => [:distance, :fuel_efficiency] do |characteristics|
              characteristics[:distance] / characteristics[:fuel_efficiency]
            end
          end

          committee :emission_factor do # kg CO2 per litre
            quorum 'from fuel', :needs => :fuel do |characteristics|
characteristics[:fuel].emission_factor
            end
          end

          committee :distance do # km
            quorum 'from annual distance',
:needs => [:annual_distance,
:active_subtimeframe] do |characteristics,
timeframe|
              characteristics[:annual_distance]
* (characteristics[:active_subtimeframe] /
timeframe.year)
            end
          end

          committee :fuel_efficiency do # km
per litre
            quorum 'default' do
              base.fallback.fuel_efficiency
            end
          end

          committee :fuel do
            quorum 'default' do
              AutomobileFuel.find_by_code 'R'
            end
          end

          committee :annual_distance do #
kilometres
            quorum 'from annual distance
estimate', :needs => :annual_distance_estimate
do |characteristics|

characteristics[:annual_distance_estimate]
            end
            quorum 'from weekly distance
estimate', :needs => :weekly_distance_estimate
do |characteristics, timeframe|

(characteristics[:weekly_distance_estimate] / 7
) * timeframe.year.days
            end

            quorum 'default' do

base.fallback.annual_distance_estimate
            end
          end

          committee :active_subtimeframe do
            quorum 'from acquisition and
retirement', :needs => [:acquisition,
:retirement] do |characteristics, timeframe|
              Timeframe.constrained_new
characteristics[:acquisition].to_date,
characteristics[:retirement].to_date, timeframe
            end
          end

          committee :acquisition do
            quorum 'from retirement',
:appreciates => :retirement do
|characteristics, timeframe|
              [ timeframe.from,
characteristics[:retirement] ].compact.min
            end
          end

          committee :retirement do
            quorum 'from acquisition',
:appreciates => :acquisition do
|characteristics, timeframe|
              [ timeframe.to,
characteristics[:acquisition] ].compact.max
            end
          end
        end
      end
    end
  end
end
```