Refaktoriseringsplan:

Skapa en ny klass som heter Position som innehåller både Direction samt koordinater i form av en Point.

Ändra namn/egenskaper för interface PassengerCar. Ett bättre namn är t.ex Transportable eftersom vi eventuellt vill lägga till ett helt annat fordon som också ska gå att transportera.

TransportTruck borde inte ändra fält hos Car. Det bryter mot Singel Responsibility Principle.

Workshops borde heller inte ändra fält hos Car. Skapa en metod i Car som sätter currentSpeed till 0 samt uppdaterar värdet loadedByWorkshop

Skapa en metod i CarController som uppdaterar spelets tillstånd, t.ex updateCars().
Vi kan skapa en helt egen separerad klass TimeListener och kalla på metoden därifrån. På så sätt kan vi ta bort beroendet mellan TimeListener och Car samt CarView.

Skapa en egen klass som endast ansvarar för att skapa instanser av bilar och bestämma deras grundvärden. Vi kan då ta bort den delen ur game-loopen i CarController. Då blir det smidigare att lägga till nya fordon utan att faktiskt ändra något i CarController.

Flytta metoden isWithinRadiusofWorkshop till den nya klassen Position och döpa om den till det mer generella namnet isWithinRadius. På så sätt kan metoden användas i olika sammanhang. Samma med carInBounds

Flytta alla metoder som hanterar bilarnas funktionalitet från CarController till en ny klass som heter VehicleManager. Det räcker med att den klassen känner till alla metoder som är unika för olika bilar och sedan göra metodanrop i CarController.

Vi kan flytta ut actionListener från CarView till CarController. På så sätt kan vi få bort det cirkulära beroendet mellan CarView och CarController