

Re: basic q about sequence diagrams

Source: <http://coding.derkeiler.com/Archive/General/comp.object/2004-03/1182.html>

From: H. S. Lahman (h.lahman_at_verizon.net)

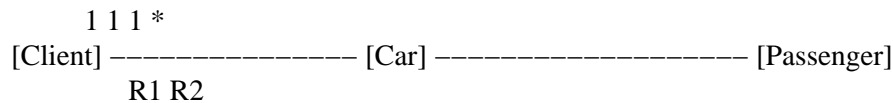
Date: 03/16/04

Date: Tue, 16 Mar 2004 21:57:59 GMT

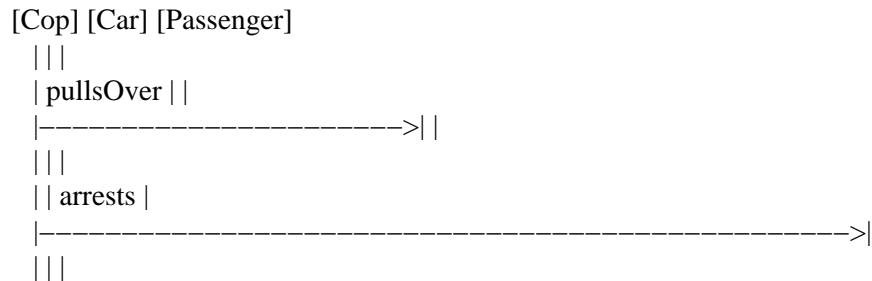
Responding to Gawnsoft...

>>If two objects are represented in a sequence diagram with one message
 >>joining them, and the message is a method call, which object is supposed to
 >>have the method as one of its members?
 >>
 >>example if I have a car object and I want to call a method
 >>car.getPassengers() to return a collection of passengers
 >>
 >>do I model it this way:
 >>
 >>[car] -----getPassengers----->[passengers]
 >>or
 >>[system] -----getPassengers----->[car]
 >>
 >>..brackets are boxes there...forgive the simplification

Let's step back to the Class Diagram. I assume we have something like:



and the Client needs to collaborate with the Car's Passengers. That is, Client sends a message directly to each Passenger. That message is what should appear in the Sequence Diagram and it will depend upon what the collaboration is. For example, substitute Cop for Client:



comp.object: Re: basic q about sequence diagrams

The `getPassengers` method is incidental to the problem solution at the OOA/D level. It simply implements relationship navigation, $R1 \rightarrow R2$, at the OOP level. Navigating that relationship path is independent of the class' semantics along the way.

[A translation code generator would provide quite generic, aspect-like code for this with attributes like "Object* R1" in [Client] and "Collection* R2" in [Car] that only a code generator could love.

Fortunately translationists rebuild the code rather than maintaining it.

B-) For manually generated code one needs the conveniences of meaningful names (`getPassengers`) and type checking during OOP.]

>

>

> *The question's really about what process you are then going to draw in the diagram.*

>

> *In this example there is a Car object, associated with a number of Passenger objects, yes?*

Yes.

>

> *A Car object keeps track of how many Passenger objects it is associated with.*

That seems like a reasonable responsibility of [Car]. However, it is likely to be delegated to a collection class that actually implements the 1:* association on the [Car] side. For example,

```
class Car
{
private:
    PtrList passengers;

public:
    getPassengerCount() {return passengers.getCount();}
}
```

where [PtrList] is the actual implementation of the relationship at the OOP level and it keeps track of the number of members it has. Again, that delegation is a convenience issue when implementing the relationship; as far as the Client is concerned the Car still owns the responsibility.

>

> *A user wants to know how many, so enquires from the system, by sending a 'how many passengers do you have?' message to the car.*

>

> *so the answer is:*

> *[system] -----getPassengers----->[car]*

Re: basic q about sequence diagrams

comp.object: Re: basic q about sequence diagrams

Taken literally, that would not be a good practice. If all the client cares about is the number of passengers, it doesn't need the passengers themselves to determine that. It should ask for only what it does need, which is a count. Which is why giving [Car] a public responsibility for knowing how many passengers it has is reasonable.

There is nothing wrong with me that could not be cured by a capful of Drano.

H. S. Lahman
hsl@pathfindermda.com
Pathfinder Solutions -- Put MDA to Work
<http://www.pathfindermda.com>
(888)-OOA-PATH