Color Modeling and the Law of Demeter

By David J. Anderson

Introduction

One of the most widely misunderstood elements of color modeling and the Domain Neutral Component is its loosely coupled nature and the use of what is often referred to as the Law of Demeter. In a DNC compliant model, it is intended that classes only hold dependencies on their immediate neighbors in the DNC. This reduces the coupling of architecture and allows decisions on the packaging and coarse-grained componentization to be postponed until much later. The DNC treats every class as a component and both its interface and degree of coupling are carefully considered.

Dependency in the DNC

We are all used to seeing the Domain Neutral Component drawn as it is in Figure 1. This diagram shows the static persistent associations between the classes. However, consider the diagram in Figure 2 which shows the dynamic dependencies in the form of <<use>> relationships between the classes in the DNC. Note how each class only holds dependencies to its immediate neighbors. This is at the heart of the agile nature of color modeling and the flexibility of the Domain Neutral Component. Building a loosely coupled functional architecture in this fashion minimizes the future impact of change and minimizes the regression effect when code needs to be refactored. It also facilitates lean software development by allowing decisions on coarse-grained componentization and the interfaces of coarse-grained components to be postponed until the last responsible moment. The result is a significant reduction in refactoring effort.

Law of Demeter

The Law of Demeter was first proposed by Ian Holland of Northeastern University, Boston, MA, in 1987. It was named after the project on which the style guide was first adopted. The law states,

Each unit should have only limited knowledge about other units: only units "closely" related to the current unit.
Or: Each unit should only talk to its friends; Don't talk to strangers.

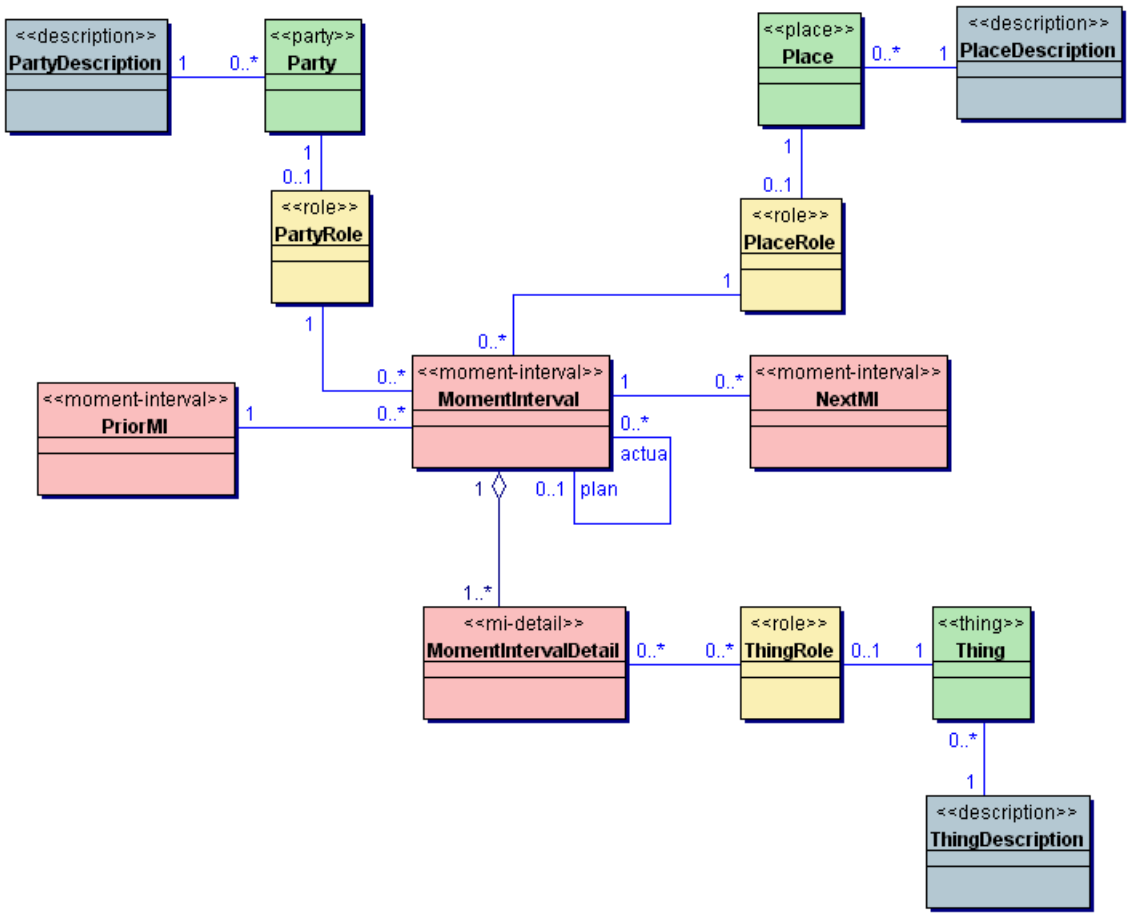

Figure 1. The Domain Neutral Component (1999)

Peter Coad widely adopted the lessons of the Law of Demeter (LoD) in his work, though he rarely referred to it by name. The trade off with LoD is that you end up with large method signatures on your classes because they spend a lot of time providing contextual message forwarding. The advantage is loose coupling – a class only knows about its neighbors. The opposite of LoD code is simply to call accessors on classes and then ask the final one in the chain, the question that you needed answering. Figure 3 shows a sequence diagram for a correctly posed question which is LoD compliant. Figure 4 shows the same query done is a typical accessor fashion which is rightly coupled and not LoD compliant.

We can see clearly that the sequence diagram in Figure 4 shows the HotelDescription class is holding dynamic dependencies to ConferenceVenue and Conference rather than its immediate neighbor Hotel. This is a tightly coupled design which may require refactoring later on, if some form of packaging or componentization is required. The design in Figure 4 is less likely to encourage reuse of the Hotel and Hotel Description classes in other applications.

Hence, reuability of an object-oriented design is affected by low-level design choices at the individual feature level. Knowing and applying the Law of Demeter as part of a coding style guide and enforcing it with design and code reviews is a major contributor to code reuse and sustained high project velocity.

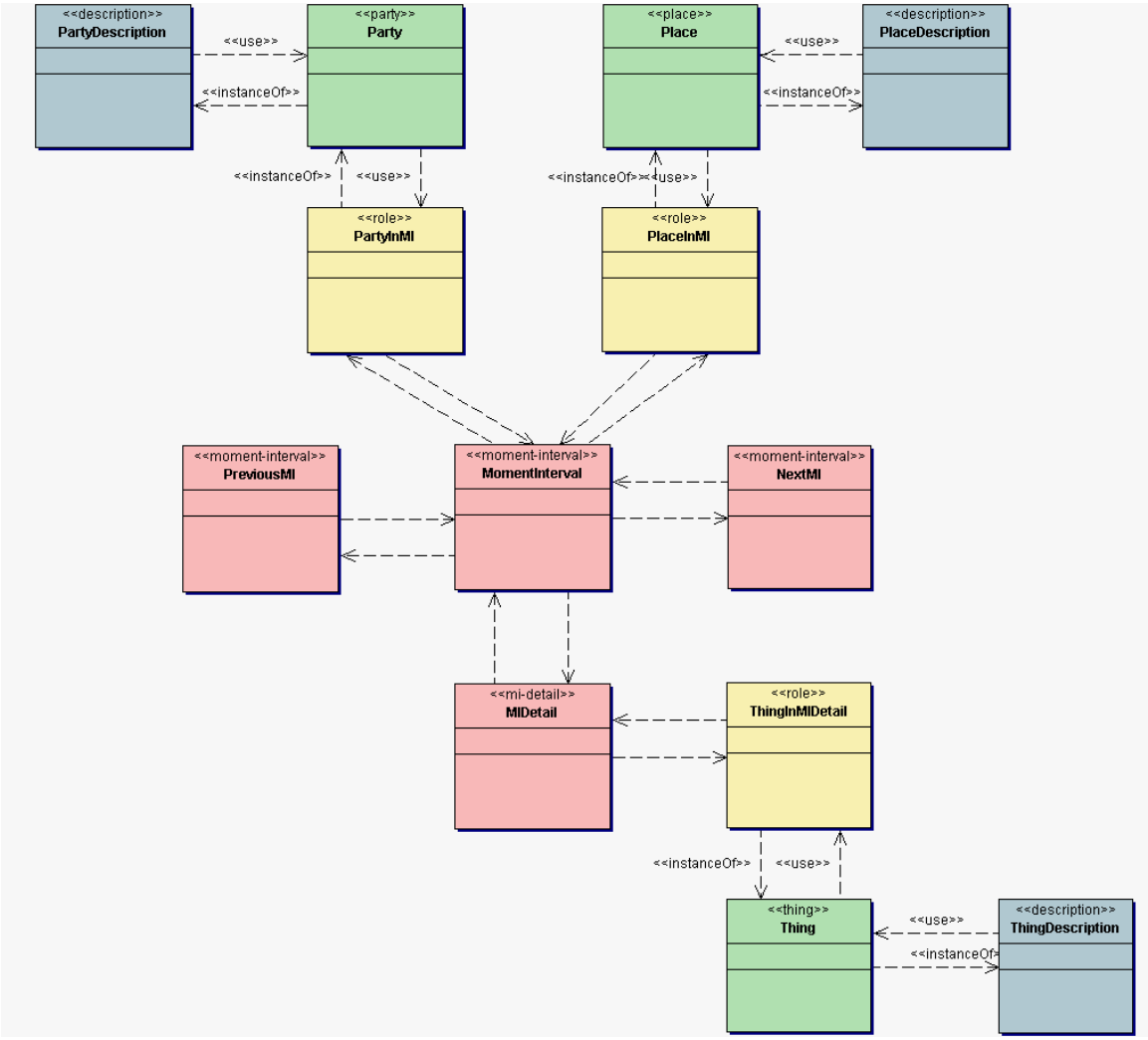

Figure 2. DNC showing Dynamic Dependencies

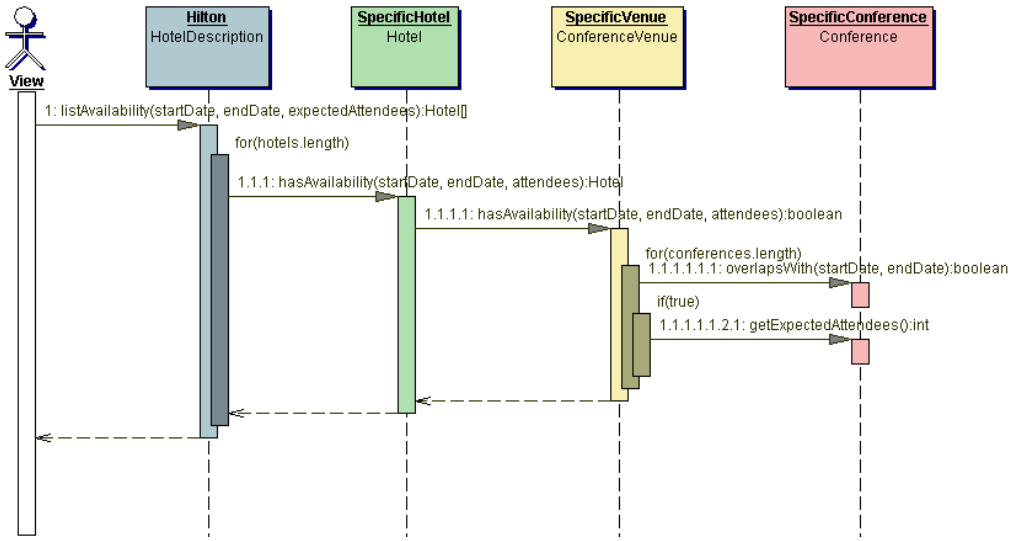

Figure 3. List hotel availability for a conference (a) in an LoD compliant manner

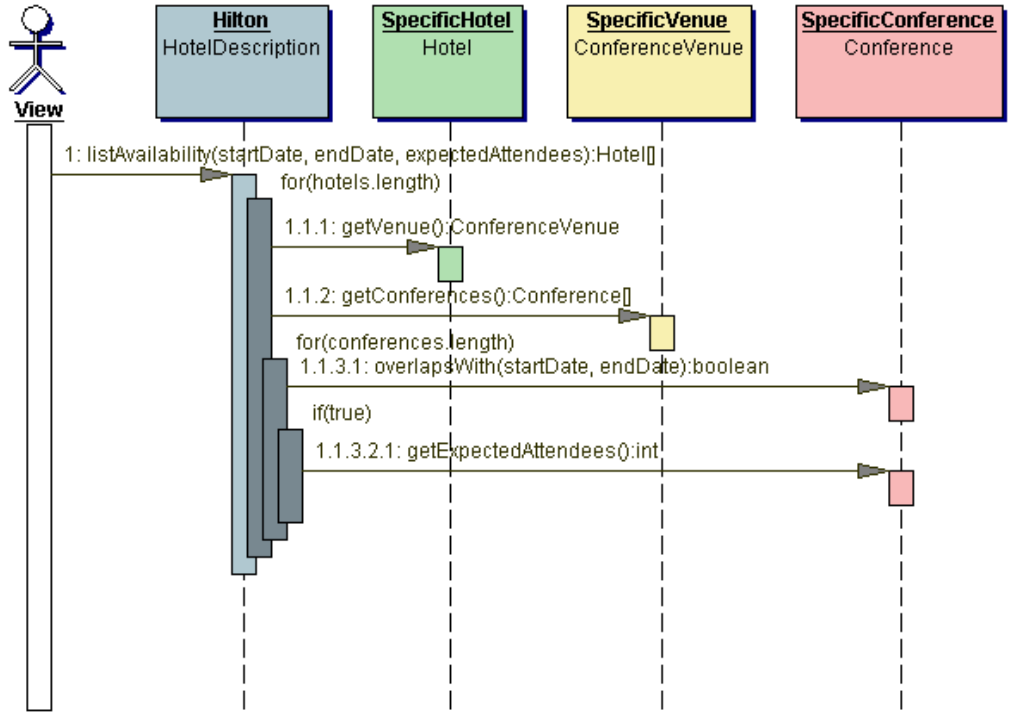

Figure 4. List hotel availability for a conference (b) in a tightly coupled manner

Summary

Color modeling with the Domain Neutral Component is a very powerful technique which quickly allows you to create a highly flexible, loosely coupled functional architecture for an object-oriented system in a language like Java. It is not well understood that the DNC makes use of the Law of Demeter and classes hold only dynamic dependencies to their immediate neighbors. This is at the heart of the flexibility and loose coupling of the DNC. Flexibility and loose coupling reduce the likelihood of refactoring becoming necessary and reduce the regression effect of refactoring when it does happen.

A loosely coupled DNC model treats each class as a component. Decisions about coarse-grained, large, distributed components or class packaging can be delayed until the last responsible moment. In the next Coad Letter, I will discuss how to componentized and package color object models.

About the author

David J. Anderson is the author of the recent book, "Agile Management for Software Engineering – Applying the Theory of Constraints for Business Results" published in Peter Coad's series by Prentice Hall PTR in September 2003. He is a program manager with Microsoft in Redmond, Washington. David was one of the team which created the popular agile development method, Feature Driven Development. He has introduced FDD at two Fortune 100 companies Sprint (a telecommunications operator in the United States) and Motorola. He holds a degree in Computer Science and Electronics from the University of Strathclyde.

David publishes new thinking regularly through his Agile Management weblog, http://www.agilemanagement.net/

Email: dja@agilemanagement.net

References

Coad Peter, Eric Le Febvre & Jeff De Luca, ***Java Modeling in Color with UML – Enterprise Components and Process***, Prentice Hall, Upper Saddle River, NJ, 1999
Karl Lieberherr, ***Law of Demeter***, Northeastern University,
http://www.ccs.neu.edu/research/demeter/demeter-method/LawOfDemeter/general-formulation.html
Palmer, Stephen R., ***A New Beginning***, The Coad Letter, 2002
http://www.thecoadletter.com/article/0,1410,29697,00.html

