Известные применения

Пример виртуального заместителя из раздела «Мотивация» заимствован из классов строительного блока текста, определенных в каркасе ET++.

В системе NEXTSTEP [Add94] заместители (экземпляры класса `NXProxy`) используются как локальные представители объектов, которые могут быть распределенными. Сервер создает заместителей для удаленных объектов, когда клиент их запрашивает. Заместитель кодирует полученное сообщение вместе со всеми аргументами, после чего отправляет его удаленному субъекту. Аналогично субъект кодирует возвращенные результаты и посылает их обратно объекту `NXProxy`.

В работе McCullough [McC87] обсуждается применение заместителей в Smalltalk для доступа к удаленным объектам. Джефри Пэско (Geoffrey Pascoe) [Pas86] описывает, как обеспечить побочные эффекты при вызове методов и реализовать контроль доступа с помощью «инкапсуляторов».

Родственные паттерны

Паттерн адаптер предоставляет другой интерфейс к адаптируемому объекту. Напротив, заместитель в точности повторяет интерфейс своего субъекта. Однако, если заместитель используется для ограничения доступа, он может отказаться выполнять операцию, которую субъект выполнил бы, поэтому на самом деле интерфейс заместителя может быть и подмножеством интерфейса субъекта.

Несколько замечаний относительно декоратора. Хотя его реализация и похожа на реализацию заместителя, но назначение совершенно иное. Декоратор добавляет объекту новые обязанности, а заместитель контролирует доступ к объекту.

Степень схожести реализации заместителей и декораторов может быть различной. Защищающий заместитель мог бы быть реализован в точности как декоратор. С другой стороны, удаленный заместитель не содержит прямых ссылок на реальный субъект, а лишь косвенную ссылку, что-то вроде «идентификатор хоста и локальный адрес на этом хосте». Вначале виртуальный заместитель имеет только косвенную ссылку (скажем, имя файла), но в конечном итоге получает и использует прямую ссылку.

Обсуждение структурных паттернов

Возможно, вы обратили внимание на то, что структурные паттерны похожи между собой, особенно когда речь идет об их участниках и взаимодействиях. Вероятное объяснение такому явлению: все структурные паттерны основаны на небольшом множестве языковых механизмов структурирования кода и объектов (одиночном и множественном наследовании для паттернов уровня класса и композиции для паттернов уровня объектов). Но имеющееся сходство может быть обманчиво, ибо с помощью разных паттернов можно решать совершенно разные задачи. В этом разделе сопоставлены группы структурных паттернов, и вы сможете яснее почувствовать их сравнительные достоинства и недостатки.

Адаптер и мост

У паттернов адаптер и мост есть несколько общих атрибутов. Тот и другой повышают гибкость, вводя дополнительный уровень косвенности при обращении