

УДК 681.3.06:534

## ПРОГРАММНОЕ СРЕДСТВО ДЛЯ МОДЕЛИРОВАНИЯ ФИЗИЧЕСКИХ ПРОЦЕССОВ ПРИ ЗАЩИТЕ РЕЧЕВОЙ ИНФОРМАЦИИ

И.В. САВЧЕНКО

*Белорусский государственный университет информатики и радиоэлектроники  
П. Бровки, 6, Минск, 220013, Беларусь*

*Поступила в редакцию 8 апреля 2015*

Приведены результаты разработки программного продукта для моделирования физических процессов при защите речевой информации на языке Java в программном комплексе Eclipse. Модель демонстрирует процесс распространения звуковых волн в помещении с учетом многократных отражений от препятствий.

*Ключевые слова:* защита речевой информации, программное средство, моделирование акустической задачи.

### Введение

В настоящее время в связи с широким применением разнообразных средств связи и информационных технологий, используемых для передачи информации, а также с опасностью ее несанкционированного перехвата, проблема защиты речевой информации приобретает особую актуальность и требует разработки новых подходов и методов к ее решению [1–6]. В этой связи одним из подходов к решению данной проблемы является моделирование условий и параметров, определяющих условия распространения речевой информации в различных средах в виде колебаний.

Для моделирования акустической задачи необходимо иметь базовую программную модель, которая может легко быть переконфигурирована для некоторого класса схожих моделей, соответствующих определенному классу акустических задач [7, 8]. В рамках каждой конкретной задачи модель должна предоставлять набор характеристик, которые можно видоизменять без вмешательства в программную реализацию самой модели. Разрабатываемая программная модель должна иметь возможность исследовать зависимость распространения речевой информации от различных физических характеристик акустической среды.

Программное средство должно отличаться универсальностью, комплексностью и доступностью для пользователя различного уровня владения навыками работы на компьютере. Для этого необходимо провести анализ предметной области, выбрать наиболее подходящую среду разработки и оптимальную архитектуру программного средства, которое в оптимальном режиме позволит моделировать акустические процессы и решать акустические задачи различной сложности и с различным набором исходных данных, что и рассматривается в рамках настоящей статьи.

Для достижения поставленной цели были определены: входные и выходные параметры базовой физической модели, критерии выбора среды разработки и непосредственно сама среда разработки программного средства, а также оптимальная архитектура программного средства.

### Параметры базовой модели

Входные величины и параметры, которыми будет оперировать программная реализация акустической модели, основываются на наборе математических формул и уравнений акустики, которые описывают физические процессы в рамках заданного класса акустических задач. К

таким процессам относятся: распространение звуковых волн в помещениях, отражение и поглощение звука различными по структуре и форме поверхностями, влияние отраженных волн на слышимость речевых или звуковых сигналов, методы управления структурой звукового поля, шумовыми характеристиками помещений и т.п.

При разработке программного средства необходимо учитывать специфику геометрической акустики, которая представляет собой предельный случай волновой акустики при стремлении длины звуковой волны к нулю, в связи с чем точность методов геометрической акустики повышается с уменьшением длины волны. Геометрическая акустика применима в случае пренебрежения волновой природой упругих колебаний и связанных с ней дифракционных явлений. Возможность применения методов геометрической акустики и точность получаемых результатов зависит от длины звуковой волны, размеров отражающей поверхности и ее расположения по отношению к источнику звука и точке приема и ряда других параметров.

Акустические волны как носители информации характеризуются следующими показателями и свойствами: энергией (мощностью), скоростью распространения в конкретной среде, величиной (коэффициентом) затухания или поглощения, условиями распространения акустической волны (коэффициентом отражения от границ различных сред, дифракцией). Эти показатели необходимо учитывать при разработке программного средства.

Однако для реализации программного средства необходимо перейти от объектов и процессов реального мира к программным абстракциям. В этой связи для расчета и визуализации распространения звуковых волн в заданном помещении в рамках программной модели выберем следующие три ключевых аспекта:

- вместо звуковых волн рассматривать звуковые лучи, в направлении которых распространяются эти волны;
- падающий и отраженный от какой-либо точки поверхности луч образует равные углы с нормалью к отраженной поверхности в этой точке;
- падающий и отраженный лучи лежат с нормалью в одной плоскости.

Принятые аспекты позволяют рассматривать распространение звуковых колебаний по аналогии с распространением световых лучей в геометрической оптике. Построение геометрических (лучевых) отражений широко применяется в архитектурной акустике, благодаря чему в рамках исследования можно опираться на теоретическую базу архитектурной акустики. При этом необходимо учитывать, что передача звука в виде голосового сигнала по воздуху возможна на очень небольшое расстояние, так как громкая человеческая речь слышна на расстоянии до 100 м. В то же время находящийся в закрытом помещении слушатель воспринимает, помимо доходящего до него прямого звука непосредственно от источника, еще и ряд его запаздывающих повторений, возникающих в результате отражения от различных поверхностей, и следующих друг за другом с малыми интервалами.

Одним из важных параметров, который определяет акустические свойства помещения, является реверберация, которая представляет собой процесс постепенного снижения интенсивности звуковых волн за счет их многократных отражений от различных препятствий. Излишне длительное затухание уменьшает четкость звучания и разборчивость речи. При короткой реверберации речь звучит глухо, утрачивает слитность и выразительность. Даже при оптимальном значении времени реверберации акустические свойства помещения могут очень различаться на различных направлениях из-за различия в путях, которые проходят отражения от источника звука до слушателя.

Все вышеназванные параметры необходимо использовать в разрабатываемой программной модели для получения наиболее точного результата при ее использовании.

### **Среда разработки и архитектура программного средства**

Критериями при выборе среды для разработки программной реализации модели являются:

- производительность готовой программы;
- возможность ее масштабирования для реализации распределенных вычислений;
- сложность реализации поставленной задачи в рамках каждой конкретной среды разработки;
- совместимость разработанного программного средства с различными операционными системами.

Исходя из перечисленных выше критериев, оптимальным языком программирования для реализации разрабатываемой модели выбран язык Java. Данный язык программирования является кроссплатформенным, позволяет достичь максимальной совместимости с любой операционной средой и легко масштабировать вычислительную систему, что в свою очередь, дает возможность оптимизировать разработанные программные средства для организации распределенных вычислений при решении поставленной задачи. Являясь языком высокого уровня, Java не требует дополнительных усилий при разработке программного средства и позволяет создавать программы в соответствии с концепциями объектно-ориентированного программирования и в рамках распространенных паттернов (или шаблонов) проектирования. В то же время одним из недостатков данного языка является более низкая производительность и большее потребление памяти в сравнении с некоторыми другими языками программирования, но этот фактор компенсируется возможностью эффективной организации распределенных вычислений.

При разработке программного средства целесообразно использовать паттерны, которые представляют собой повторяемую архитектурную конструкцию. Шаблоны, в свою очередь, делятся на идиомы, которые учитывают специфику конкретного языка программирования, и архитектурные шаблоны, которые охватывают архитектуру всей программной системы и не находятся в зависимости от конкретного языка программирования, что дает возможность использовать их для абстрактного проектирования программного средства. Также архитектурные шаблоны дают больше возможностей для повторного их использования и унификации процесса разработки. В этой связи в рамках поставленной задачи с учетом допущений, принятых выше, проведено исследование различных решений для реализации программной архитектуры разрабатываемого приложения.

Исходя из практических целей использования, разрабатываемое программное средство должно быть легко адаптируемым к новым задачам, что создает необходимость заложить возможность частого изменения интерфейсов и конкретных реализаций, используемых в программном средстве. Для этого необходимо четко разделить используемые абстракции и реализации, в связи с чем оптимальным выбором для данного программного средства является шаблон проектирования «Мост». При реализации базовой архитектуры приложения через шаблон «Мост», изменение структуры интерфейса не мешает изменению структуры реализации. На рис. 1 изображена диаграмма, описывающая данный шаблон.

Исходя из возможностей выбранного паттерна, для программы моделирования распространения звуковых волн в помещении были выбраны следующие абстракции, соответствующие компонентом UML диаграммы:

- Client – пользовательский интерфейс и соответствующий ему набор исходных параметров и значений, которые далее будут обработаны для получения выходных данных;
- Abstraction – компонент, который работает с входными и выходными данными и алгоритмами через интерфейс Implementor;
- Implementor – интерфейс, которому соответствует конкретная реализация модели ConcreteImplementorA или ConcreteImplementorB;
- ConcreteImplementorA и ConcreteImplementorB – конкретные реализации модели;
- Refined abstraction расширяет интерфейс, определенный сущностью «Abstraction».

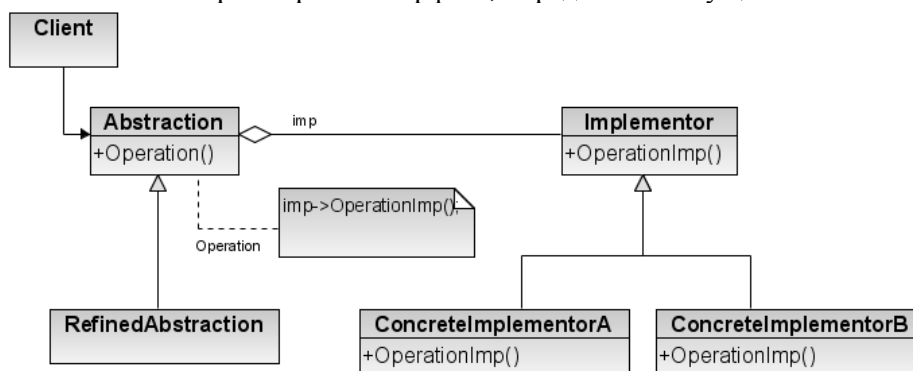


Рис. 1. Диаграмма, описывающая шаблон проектирования «Мост»

Минимальной абстракцией, используемой в приложении, выбран звуковой луч, вдоль которого распространяются звуковые волны. С данной абстракцией связан набор свойств для луча: начальная и конечная точки, начальная амплитуда колебаний, начальная фаза и т. д. Также заданная абстракция позволяет получать производные физические величины от заданного набора свойств и величины, изменяющиеся во времени: амплитуда в заданной точке с учетом затуханий, фаза в заданной точке и др. Чтобы абстракция могла использоваться в иных задачах либо для расширения изначальной задачи, необходимо изначально реализовать механизм расширения абстракции и добавления дополнительных свойств к уже имеющимся.

В данном случае оптимальным решением для реализации поставленных задач является использование структурного шаблона проектирования «Декоратор», который служит для динамического подключения дополнительного поведения к объекту, который представляет абстракция в рамках программной реализации. Шаблон «Декоратор» предоставляет собой гибкую альтернативу практике создания подклассов с целью расширения функциональности и предусматривает расширение функциональности объекта без определения подклассов. Реализация паттерна заключается в создании абстрактного класса, представляющего как исходный класс, так и новые, добавляемые в класс функции. В классах-декораторах новые функции вызываются в требуемой последовательности – до или после вызова последующего объекта. В то же время остается возможность использовать исходный класс (без расширения функциональности), при условии, что на его объект сохранилась ссылка.

К достоинствам использования выбранного паттерна для реализации расширяемой абстракции относятся: реализация добавляемой функциональности в небольших объектах, возможность динамического добавления данной функциональности до или после основной функциональности объекта, возможность избегать перегрузки функциональными классами на верхних уровнях иерархии и отсутствие идентичности декоратора и его компонент.

На рис. 2 представлена диаграмма описанного паттерна, на которой отображены основные его компоненты и взаимосвязи между его компонентами, которые затем будут транслированы в лексические конструкторы выбранного языка программирования, реализующие описанные взаимосвязи.

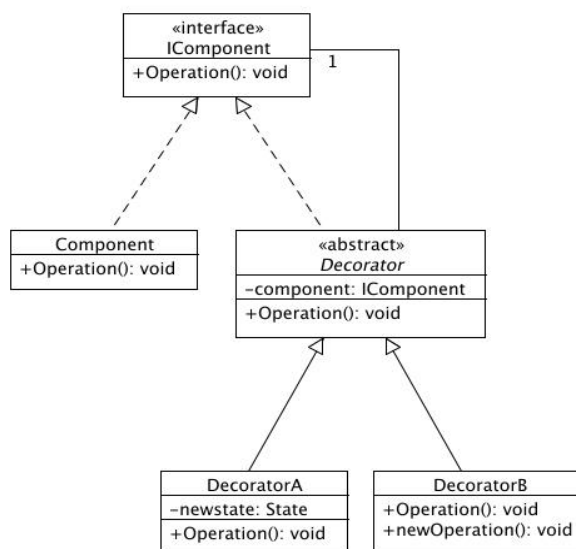


Рис. 2. Диаграмма описанного паттерна

Для разрабатываемой программы необходимо выбрать абстракции, соответствующие каждому из компонентов диаграммы. Базовый класс паттерна «Декоратор» в данном случае соответствует минимальной реализации звукового луча. Минимальным набором свойств в данном случае является начальная и конечные точки. Чтобы унифицировать программную архитектуру, каждое новое свойство сверх минимальной реализации добавляется через паттерн «Декоратор». Исходя из этого, существуют следующие абстракции, соответствующие каждому компоненту UML диаграммы:

– IComponent – представляет собой интерфейс, описывающий поведение, соответствующее минимальной реализации звукового луча и позволяющий построить прозрачную в рамках дальнейшей разработки иерархию классов. Это дает возможность итерационно развивать программное средство и свободно добавлять новый функционал на любом этапе разработки;

– Component – является непосредственной реализацией базового интерфейса, который определяет поведение компонента в рамках базового интерфейса, инкапсулируя программную логику;

– Decorator – представляет собой абстрактный класс, который позволяет расширять функционал базового класса. С целью расширения функционала базового класса Decorator реализует интерфейс IComponent, чтобы оставаться совместимым с методами сторонних классов, которые взаимодействуют с экземпляром объекта через интерфейс IComponent. Также данный абстрактный класс реализует логику работы с вложенными экземплярами классов, реализующими интерфейс IComponent, чей функционал будет расширен с помощью классов, унаследованных от Decorator. Обязательным условием для вложенного класса является то, что он должен реализовывать интерфейс IComponent;

– DecoratorA, DecoratorB, использующиеся для реализации абстрактного класса Decorator, в каждый из которых добавлен независимый набор дополнительных свойств для звукового луча.

Таким образом, паттерны позволяют перейти от постановки задачи к непосредственной реализации архитектуры программного средства и рассмотрению ключевых компонентов с последующей реализацией программной логики.

На основании разработанной архитектуры был реализован прототип программного средства, демонстрирующий работу описанного выше алгоритма в рамках выбранной математической модели.

В качестве входных параметров были заданы:

- размер помещения и взаимное расположение стен;
- источник звука, являющийся центром исходящих лучей;
- точность расчета, которая включает в себя количество лучей и количество отражений.

При работе с программой изменялись значения, определяющие точность заданных параметров, чтобы итерационно продемонстрировать принцип работы алгоритма.

Работа программы при 32 лучах без отражений определяет радиальное, относительно источника звука, распространение звуковой волны с линейным изменением плотности звука в зависимости от расстояния от источника звука до препятствия. Построение распространения звуковой волны при заданном единичном отражении для каждого луча, в результате чего формируются вторичные звуковые волны, вносят неоднородность в плотность звуковых волн внутри изучаемого помещения. При этом данная неоднородность обусловлена погрешностью моделирования и не должна учитываться при анализе данных. Изображение (рис. 3) демонстрирует двойные отражения при прежних начальных условиях.

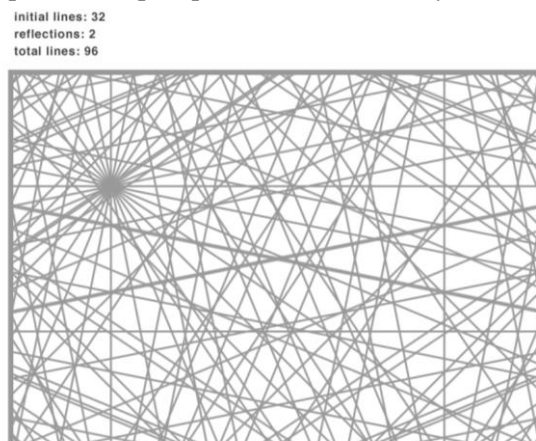


Рис. 3. Радиальное распространение звуковой волны при 32 лучах и заданном двойном отражении для каждого луча

Неоднородность значения плотности звуковых волн уже в меньшей степени обусловлена погрешностью и ближе к реальным значениям, которые могут быть получены в ходе лабораторных экспериментов.

Используемый алгоритм обладает линейной сложностью относительно используемых ресурсов: память и процессорное время. В обоих случаях сложность алгоритма будет определяться формулой  $d = O(lCount * rCount)$ , где  $lCount$  – количество лучей, определенных для источника звука,  $rCount$  – количество отражений для каждого луча.

При определении сложности алгоритма не учитывались такие факторы, как оптимизация алгоритма для одного из ресурсов, оптимизация для масштабирования вычислительной системы и оптимизация непосредственно компилятора.

### **Заключение**

Таким образом, для решения акустических задач проведена адаптация физической модели под программную реализацию, для чего определены входные и выходные параметры базовой физической модели, критерии выбора среды разработки и непосредственно сама среда разработки программного средства, а также предложена архитектура программного средства.

Для определения входных и выходных параметров базовой физической модели выбраны паттерны для реализации физических процессов и сущностей. С учетом требований к программной реализации и исследуемой физической модели, выбран язык программирования Java, который позволяет достичь максимальной совместимости разрабатываемого программного средства с любой операционной средой и создавать программы в соответствии с концепциями объектно-ориентированного программирования и в рамках распространенных паттернов проектирования. Предложенная архитектура программного средства представлена в виде оптимального набора шаблонов проектирования, которые представляют собой абстрактное описание архитектурной конструкции для решения распространенных проблем проектирования и позволяют масштабировать программное средство, усложнять отдельные компоненты, при этом не усложняя связи и взаимодействия между компонентами.

## **SOFTWARE TOOLS FOR MODELING OF PHYSICAL PROCESSES IN THE PROTECTION OF THE SPEECH INFORMATION**

I.V. SAVCHENKO

### **Abstract**

A version of software for the simulation of physical processes in the protection of the speech information in the Java language in the software package Eclipse is shown. The simulated system consists of a set of classes and patterns generalizing common approaches to solving problems of acoustic beam method. The model demonstrates the process of propagation of sound waves in a room with a view of multiple reflections from obstacles.

### **Список литературы**

1. Барсуков В.С. Безопасность: технологии, средства, услуги. М., 2001.
2. Герасименко В.А., Малюк А.А. Основы защиты информации. М., 1997.
3. Дворянkin С.В. Макаров Ю. К., Хорев А. А. // Защита информации. Инсайд. 2007. № 2. С. 18–25.
4. Дворянkin С.В. Компьютерные технологии защиты речевых сообщений в каналах электросвязи. М., 1999.
5. Железняк В.К. Макаров Ю.К., Хорев А.А. // Специальная техника. 2000. № 4. С. 39–45.
6. Хорев А.А. // Специальная техника. 2006. № 6. С. 53–61.
7. Соммервил И. Инженерия программного обеспечения. М., 2002.
8. Filippi P., Habault D., Lefebvre J-P. et.al. Acoustics: Basic physics, theory and practice. Paris, 1999.