УДК 681.3.06

СОЗДАНИЕ РАСПРЕДЕЛЕННЫХ ПРИЛОЖЕНИЙ НА БАЗЕ ТЕХНОЛОГИИ УДАЛЕННОГО ДОСТУПА

к.т.н. В.В. Калачёва, к.т.н. С.В. Дуденко
(представил д.т.н., проф. Ю.В. Стасев)

Исследуются вопросы применения технологии удаленного доступа Java/RMI для создания прикладного программного обеспечения распределенных информационных систем произвольного назначения на основе объектно-ориентированного программного моделирования.

Введение. Возросший в последнее время интерес к созданию распределенных приложений объясняется следующими обстоятельствами [1, 2]:

– обеспечивается одновременное качественное обслуживание как малого, так и очень большого количества клиентов;

– гарантируется устойчивость к ошибкам пользователей и сбоям в системе коммуникаций за счет использования транзакций;

– допускается непрерывной круглосуточный режим работы в течение недель и месяцев;

– гарантируется высокий уровень безопасности приложения за счет контроль доступности к ресурсам системы, защищенности информации на всех этапах функционирования и отслеживания выполняемых операций;

– реализуется высокая скорость разработки приложений, простота их сопровождения и модификации программистами средней квалификации.

Целью данной работы является выбор технологии, позволяющей создавать сложные распределенные приложения.

Возможности технологии удаленного доступа Java/RMI. Технология RMI позволяет создавать сложные распределенные приложения. RMI обладает всеми преимуществами языковой среды Java, обеспечивает объектно-ориентированное программирование, гарантирует высокий уровень безопасности и надежности, многопотоковость, многоплатформенность, независимость от операционной системы. Технология Java/RMI в Internet-приложениях считается лучшим решением для работы с удаленными объектами.

RMI может работать поверх протокола IIOP, что позволяет связываться с CORBA-объектами. Протокол IIOP способен передавать данные различных типов, включая структуры и объединения, в том числе со-

держащие рекурсивные определения. Важная особенность RMI заключается в том, что он представляет программируемый интерфейс для работы с сетями в отличие от сокетов TCP. Главное преимущество его в том, что он предлагает интерфейс более высокого уровня, основанный на вызовах методов, так как если бы удаленный объект обрабатывался локально. RMI более удобен и более естественен, чем интерфейс, основанный на сокетах, но он требует выполнения Java-программ на обоих концах соединения. Сетевое соединение, тем не менее, достигается использованием все того же TCP/IP протокола. RMI может быть применен для создания распределенных клиент-серверных приложений трехуровневой архитектуры на языке Java.

Реализация распределенных приложений по технологии Java/RMI. Удаленный объект в Java представляет собой экземпляр класса, который реализует интерфейс Remote (удаленный интерфейс) из пакета java.rmi. Интерфейс Remote может быть расширен включением в него дистанционно разделяемых методов. Удаленный метод в java – это метод интерфейса реализуемого интерфейс Remote.

Удаленный интерфейс имеет следующие характеристики:

– удаленный интерфейс должен быть объявлен как public. В противном случае клиент получит ошибку, когда будет пытаться загрузить удаленный объект, который описывает удаленный интерфейс, если клиент не будет в том же самом пакете что и удаленный интерфейс. Удаленный интерфейс расширяет java**.**rmi**.**Remote интерфейс;

– каждый метод интерфейса должен быть объявлен с исключением java**.**rmi**.**RemoteException, которое активизируется всякий раз, когда метод удаленного вызова дает сбои;

– тип данных любого удаленного объекта, который передается как аргумент или возвращаемое значение должен быть таким же, как и тип в удаленном интерфейсе.

Обычно распределенное RMI-приложение на Java состоит из клиента и сервера. Типичное серверное приложение создает распределенные объекты, делая ссылки на них доступными, и ждет, когда клиенты вызовут методы этого объекта. Типичное клиентское приложение получает удаленную ссылку на один или несколько удаленных объектов и запускает их методы. RMI предоставляет механизм, благодаря которому клиент и сервер соединяются и передают информацию один другому.

Распределенное приложение нуждается в определении места расположения удаленного объекта: Приложения могут использовать один из двух механизмов для получения ссылки на удаленные объекты. Приложение может зарегистрировать свой удаленный объект с помощью простого

RMI средства – rmiregistry, или приложение может пересылать и возвращать ссылку на удаленный объект как часть его обычной операции. Деталями обмена информацией с удаленными распределенными объектами управляет RMI; для программиста удаленная коммуникация выглядит так же как и стандартный вызов методов. Поскольку RMI позволяет вызывающей программе пересылать данные и методы в удаленный объект, то RMI представляет необходимый механизм для загрузки кода объекта и передачи его данных.

Распределенное приложение RMI может использовать реестр (registry-программа, выполняющая функцию сервера регистрации удаленного объекта) для получения ссылки на удаленный объект. Сервер вызывает registry для того, чтобы ассоциировать (или связать) имя с удаленным объектом. Клиент ищет удаленный объект по его имени в регистре, который находится на сервере, чтобы вызвать методы полученного объекта.

Распределенное приложение по технологии Java/RMI может быть построено из интерфейсов и классов. Интерфейсы определяют методы, а классы реализуют методы, определенные в интерфейсах и, возможно, расширяют эти методы. В распределенном приложении некоторые из описаний хранятся на различных виртуальных машинах. (Объекты, методы которых могут вызываться на разных виртуальных машинах – удаленные объекты.)

RMI интерпретирует удаленный объект отлично от не удаленного: объект пересылается от одной виртуальной машине другой. Вернее, производится копирование описания объекта в получающую виртуальную машину, **RMI** пересылает удаленный stub (заглушку) к удаленному объекту. Stub действует как локальный представитель для удаленного объекта, и он (**stub**) по существу представляет собой удаленную ссылку для вызывающего приложения.

Заглушка, для удаленных описаний объекта, подобна по характеру интерфейсам, которые реализуются удаленным объектом. Это позволяет заглушке быть преобразованной к любому из интерфейсов, которые удаленный объект реализует. Однако, это так же означает, что только эти методы, определенные в удаленном интерфейсе, могут быть вызваны в виртуальной машине получателя. Процедура разработки распределенного RMI-приложения включает следующие этапы.

1. Создание компонентов распределенного приложения. Этот процесс можно разделить на три шага:

Определение удаленных интерфейсов. Удаленный интерфейс описывает методы, которые могут быть удаленно вызваны клиентом. Клиентские программы не реализуют классы удаленных интерфейсов. Часть

разработки таких интерфейсов представляет собой определение всех локальных объектов, которые будут использоваться как параметры и возвращать значения этих методов. Если любой из этих интерфейсов или классов еще не существует, необходимо их определить. В нашем случае имя удаленного интерфейса – Server. Он описывает единственный метод – sayHello, который будет удаленно вызван клиентами.)

Реализация удаленных объектов (серверов). Удаленные объекты должны реализовывать один или более удаленных интерфейсов. Удаленный класс объекта может включать реализации других интерфейсов (локальных или удаленных) и другие методы (которые доступны только локально). Если любые локальные классы используются как параметры или возвращаемые значения для любого из этих методов, то они должны быть реализованы соответствующим образом. Реализуемым объектом является экземпляр класса *ServerImpl*. *ServerImpl* реализует единственный удаленный интерфейс – Server, реализует определенный в интерфейсе метод.

Реализация клиентов. Клиенты, которые используют удаленные объекты, могут быть описаны в любое время, после того как определен удаленный интерфейс и после того, как удаленные объекты были объявленными доступными из сети.

2. Компиляция исходных кодов и генерация заглушек. Это процесс можно разделить на два шага. На первом шаге используется компилятор javac для компиляции исходных файлов, которые содержат реализацию удаленного интерфейса, реализации классов сервера и классов клиента. На втором шаге используется компилятор rmic для создания заглушек и скелетонов удаленных объектов. RMI использует заглушки и скелетоны классов удаленных объектов, как представителей (делегатов) в клиентах. Таким образом, клиенты могут соединяться с удаленным объектом.

3. Сделать классы доступными в сети. Для этого необходимо запустить программу rmiregistry, (входящую в состав jdk) как отдельный процесс. Данная программа представляет rmi-реестр, в котором зарегистрированы интерфейсы, заглуши и другие классы, которые будут загружаться клиентом. RMIregistry для регистрации необходимых классов использует переменную среды CLASSPATH, поэтому все классы/интерфейсы, которые подлежат регистрации, необходимо прописать в этой переменной.

4. Запуск приложения. Для запуска приложения необходимо:

– чтобы на клиентской машине находилась клиентская часть приложения, а именно файл **Client.java** и заглушка **ServerImpl_stub**, которая является делегатом от клиента – с помощью нее происходит удаленное соединение с серверной частью приложения данного приложения;

– чтобы были выполнены следующие команды: **java** ServerImpl –

запуск сервера; **java** Clinet – запуск клиента.

Ниже (рис. 1) предоставляется UML – диаграмма (диаграмма классов).

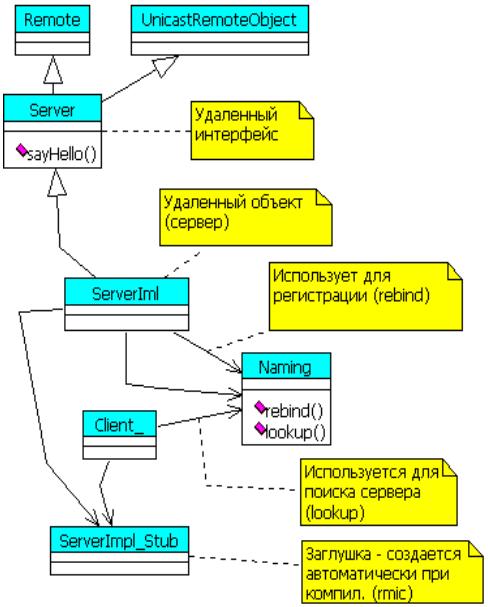

Рис. 1. Диаграмма классов

Выводы. Таким образом, технология RMI позволяет создавать сложные распределенные приложения, обладает всеми преимуществами языковой среды Java, обеспечивает объектно-ориентированное программирование, гарантирует высокий уровень безопасности и надежности, многопотоковость, многоплатформенность, независимость от операционной системы. Технология RMI позволяет задавать удаленные интерфейсы, методы которых могут вызываться через сеть. С помощью специального компилятора можно создавать объекты-заглушки, которые позволяют не только получать ссылку на удаленный объект, но и иметь полную копию объекта. Технология Java/RMI в Internet-приложениях считается лучшим решением для работы с удаленными объектами.

ЛИТЕРАТУРА

1. *Кульба В.В., Ковалевский С.С., Косяченко С.А., Сиротюк В.О. Теоретические основы проектирования оптимальных структур распределенных баз данных. Серия «Информатизация России на пороге XXI века». – М.: СИНТЕГ, 1999. – 660 с.*
2. *Ершов В.П., Кузнецов Н.А. Мультисервисные телекоммуникционные сети. – М.: МГТУ им. Н.Э. Баумана, 2003. – 428 с.*
3. *Мамиконов А.Г., Кульба В.В., Косяченко С.А., Ужастов И.А. Оптимизация структур распределенных баз данных в АСУ. – М.: Наука, 1990. – 240 с.*

Поступила 22.09.2004

***КАЛАЧЁВА Вероника Валерьевна**, канд. техн. наук, старший научный сотрудник НИЛ ХУ ПС. В 1995 году окончила ХГТУРЭ. Область научных интересов – системы поддержки принятия решений в логистических информационных системах.*

***ДУДЕНКО Сергей Васильевич**, канд. техн. наук, старший преподаватель кафедры*

ХВУ. В 1995 году окончил ХВУ. Область научных интересов – способы и средства безопасной передачи данных в информационно-вычислительных сетях.