Реинженеринговая технология автоматизированного построения распределенных вычислительных систем на основе автономно разработанных фортранных программ

А.П.Сапожников, Т.Ф. Сапожникова
Лаборатория информационных технологий, ОИЯИ

В последнее десятилетие XX века в мире персональных компьютеров началось бурное развитие объектно-ориентированных технологий программирования на основе языков C++ и Паскаль. Сейчас уже невозможно представить себе солидную прикладную систему, работающую в стиле MS DOS, т.е. без современной оконной графики и "мышки". В то же время основная масса программ численных расчетов, лежащих в основе большинства прикладных систем, пришла в мир ПК еще из древней эпохи больших (по размеру!) компьютеров. Практически это означает, что программы написаны на Фортране.

Статическая природа Фортрана не позволяет ему стать инструментом объектно-ориентированного программирования. К тому же этот язык уже почти не развивается, а новые поколения программистов предпочитают C++ и Delphi. С другой стороны, трудоемкость программирования алгоритмов численных расчетов практически не зависит от используемого языка. Переписать серьезную программу с Фортрана на C как правило, способен только человек, хорошо разбирающийся в алгоритме, т.е. автор, а он скорее всего уже далек от активного программирования. Использование же конверторов вроде известного F2C [1] абсолютно не улучшает ситуацию, поскольку:

- программа как была "черным ящиком", так им и остается;
- пока не известны конверторы с Фортрана в Паскаль и C++.

Поэтому можно предположить, что большие вычислительные программы, реализованные в свое время на Фортране, вынужденно останутся в своем фортранном виде по крайней мере в ближайшее обозримое время.

Другой важной особенностью современного программирования является тенденция к распределению вычислений между несколькими, в общем случае различными компьютерами вычислительной сети. Существующие стандартные технологии распараллеливания программ, такие как MPI [2] и OpenMP [3], ориентированы на разбиение вычислительной задачи на более мелкие процессы. Как правило, это разбиение осуществляется вручную при разработке или модернизации программного обеспечения. В то же время технологии объединения готовых вычислительных блоков в более крупные распределенные системы практически отсутствуют.

Предлагается реинженеринговая технология автоматизированного построения распределенных вычислительных систем на основе автономно разработанных фортранных программ [4]. Архитектура распределенной системы достаточно нетрадиционная. Система состоит из единственного клиента и одного или нескольких вычислительных серверов, работающих либо прямо на клиентской рабочей станции, либо на удаленных компьютерах, входящих в состав локальной сети (рис. 1). Каждый такой сервер - это процесс, исполняющий конкретную вычислительную программу. Количество серверов и их местонахождение определяются потребностями клиента.

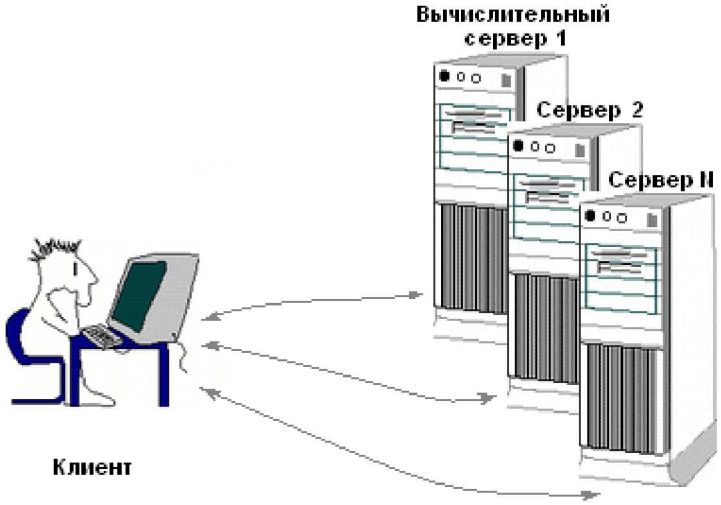

Рис. 1: Организация распределенной системы для решения больших вычислительных задач.

Ключевым моментом является идея автоматического построения вычислительного сервера из готовых, автономно разработанных фортранных программ методом их автоматического реинжениринга, исключающего ручное преобразование исходных текстов. Этим достигается возможность интеграции старых, проверенных временем программ, созданных предыдущими поколениями разработчиков, в состав более крупных современных систем обработки информации, включающих развитые средства визуализации, базы данных и прочие механизмы человеко-машинного общения.

Целью сервера является решение конкретной вычислительной задачи, поставленной клиентом. В процессе решения сервер может потреблять информацию как из автономно подготовленных файлов данных, так и в диалоге с клиентом. Необходимые в ходе решения задачи операции ввода-вывода интерпретируются клиентом, но реализуются в сервере с помощью традиционных фортранных операторов READ и WRITE. Заявки на выполнение этих операций подаются по инициативе сервера.

Рабочая станция пользователя содержит программу-клиент, распределяющую работы между независимыми вычислительными серверами. Кроме запуска вычислительных серверов и исполнения их запросов на ввод-вывод клиентская программа (и только она) осуществляет интерактивное общение с пользователем. Такая организация позволяет полностью отделить этап решения вычислительной задачи от этапа интерпретации полученных результатов, что в свою очередь позволяет использовать на каждом из этапов как наиболее подходящие инструментальные средства, так и независимые друг от друга коллективы разработчиков.

Разработан и проверен на большом объеме тестового материала конвертор фортранных программ F2F, осуществляющий преобразования исходных текстов вычислительных программ, необходимые для их работы в составе вычислительного сервера. Преобразованные программы работают в отдельном адресном пространстве, в общем случае, на удаленном компьютере. Это дополнительно повышает надежность всей системы, поскольку возможные ошибки на клиентской стороне не могут повлиять на ход выполнения программ вычислительного сервера. F2F организует необходимое для взаимодействия с клиентом программное окружение сервера. Операторы ввода-вывода в исходном тексте программы заменяются на обращения к подпро-

граммам, реализующим протокол взаимодействия с клиентом. В эти подпрограммы встраивается информация о текущем выполняемом операторе ввода-вывода: текст оператора, номер его строки в исходной программе, имя текущего файла данных, участвующего в обмене, и номер записи. Это позволяет при ошибках ввода-вывода получить максимально подробную информацию о месте возникновения ошибки, поэтому преобразованная программа становится даже в чем-то лучше исходной. Кроме того, F2F заменяет операторы OPEN открытия файлов данных на вызов подпрограммы, которая может осуществлять поиск файлов не только в конкретно указанном справочнике (директории), но и в независимо заданном списке справочников. Во все операции ввода-вывода F2F встраивает перехват возможных ошибок, что позволяет исключить неконтролируемые "зависания" программ вычислительного сервера. Еще одной привлекательной возможностью F2F является умение встраивать в конвертируемую программу команды выдачи трассировочной информации о вызовах подпрограмм оператором CALL, что весьма полезно при поиске ошибок в вычислительной программе.

Целью конвертора F2F является превращение программ, предназначавшихся для взаимодействия непосредственно с человеком, в программы, взаимодействующие с процессом-клиентом и работающие в составе распределенной системы. По существу F2F является транслятором с Фортрана на Фортран. Именно он и является основным инструментальным средством реинжиниринга в нашем проекте.

Несмотря на то, что F2F вынужден проделывать весьма глубокий синтаксический анализ конвертируемой программы, скорость его работы не менее, чем скорость работы фортранного компилятора. Первоначально мы предполагали, что конвертор F2F будет работать в полуавтоматическом режиме, беря на себя только основную часть рутинной работы по преобразованию операторов ввода-вывода, синтаксис которых в языке Фортран весьма сложен. В отдельных случаях допускалась помощь со стороны человека. Однако со всеми синтаксическими проблемами удалось успешно справиться. Сейчас F2F практически не требует ручной доработки получаемой им программы. Его работоспособность в автоматическом режиме проверена на реальных вычислительных программах, состоящих из сотен тысяч строк текста.

Разработан программный интерфейс между вычислительным сервером и программой-клиентом. Использование этого интерфейса и составляет обязательную основу предлагаемой технологии построения распределенных программных систем. Оговоримся, что эта технология вовсе не требует построения серверной части системы именно из готовых программ. Если вычислительный сервер разрабатывается заново, то у разработчика появляется выбор:

1. сделать программу "в старом стиле", используя привычные операторы ввода-вывода, а потом конвертировать ее;

2. использовать собственные инструментальные средства с соблюдением необременительных требований интерфейса.

Такой подход, в частности, позволяет привлекать к разработке вычислительных серверов опытных специалистов по численным методам, не желающих выходить за рамки привычного им Фортрана.

Коммуникации клиент-сервер осуществляются в рамках COM-технологии. Единицей обмена при этом является строка текста. Это позволяет локализовать все об-

мены в единственном месте и упрощает маршаллинг данных. При кажущейся ограниченности этого интерфейса он вполне достаточен, т.к. со стороны сервера в обмене участвует сугубо текстовая информация, а на стороне клиента необходимые форматные преобразования необременительны. В ходе проектирования коммуникационного уровня выяснилось, что традиционная модель взаимодействия клиента и сервера, когда подавший очередной запрос клиент дожидается окончания отработки этого запроса сервером, весьма неудобна для организации именно больших вычислительных задач: одна из сторон всегда вынуждена ждать. Мы использовали практически неопубликованную в отечественной литературе методику взаимодействия с использованием механизма обратных вызовов (CallBack) клиента сервером. При этом серверу предоставляется возможность своевременно получить у клиента необходимую для работы информацию, а по окончании работы сервер просто заканчивает свою деятельность, освобождая машинные ресурсы. Клиент же, в промежутках между отработкой сравнительно редких обратных вызовов, может заниматься своей основной деятельностью: интерактивным общением с пользователем, например, визуализацией получаемых от сервера результатов. Кроме того, такой подход позволяет клиенту взаимодействовать одновременно с несколькими независимо запущенными серверами, практически не усложняя программирование ни со стороны серверов, ни со стороны клиента.

Таким образом, возникает новая, простая и симметричная модель взаимодействия клиента и сервера, состоящая всего из двух базовых запросов:

1. **RequestFromClient** запускает вычислительный сервер, поручая ему конкретную работу;

2. **RequestFromServer** запрашивает у клиента дополнительную информацию.

Оба эти базовых запроса оперируют только текстовой информацией, что существенно упрощает интерфейс. Возникающая при этом необходимость в форматных преобразованиях для вычислительных задач не является обременительным ограничивающим фактором.

Итак, предлагается реинженеринговая технология автоматизированного построения распределенных вычислительных систем на основе готовых, автономно разработанных фортранных программ. Существенными чертами этой технологии являются:

1. использование автоматического конвертора F2F для получения программы вычислительного сервера;

2. построение клиентского приложения на базе готового программного обеспечения типового клиента.

Для иллюстрации методики построения вычислительного сервера из автономно разработанных программ с помощью F2F-технологии использовалась **MINUIT** [5] - программа минимизации функции многих переменных. Фортранный текст без каких-либо изменений был обработан конвертором F2F, после чего все общение MINUIT с внешним миром переключилось на достаточно простую клиентскую программу. Эта программа, написанная на Delphi, занималась интерпретацией запросов на ввод-вывод и визуализацией процесса минимизации, изначально в MINUIT не предусмотренной (рис. 2).

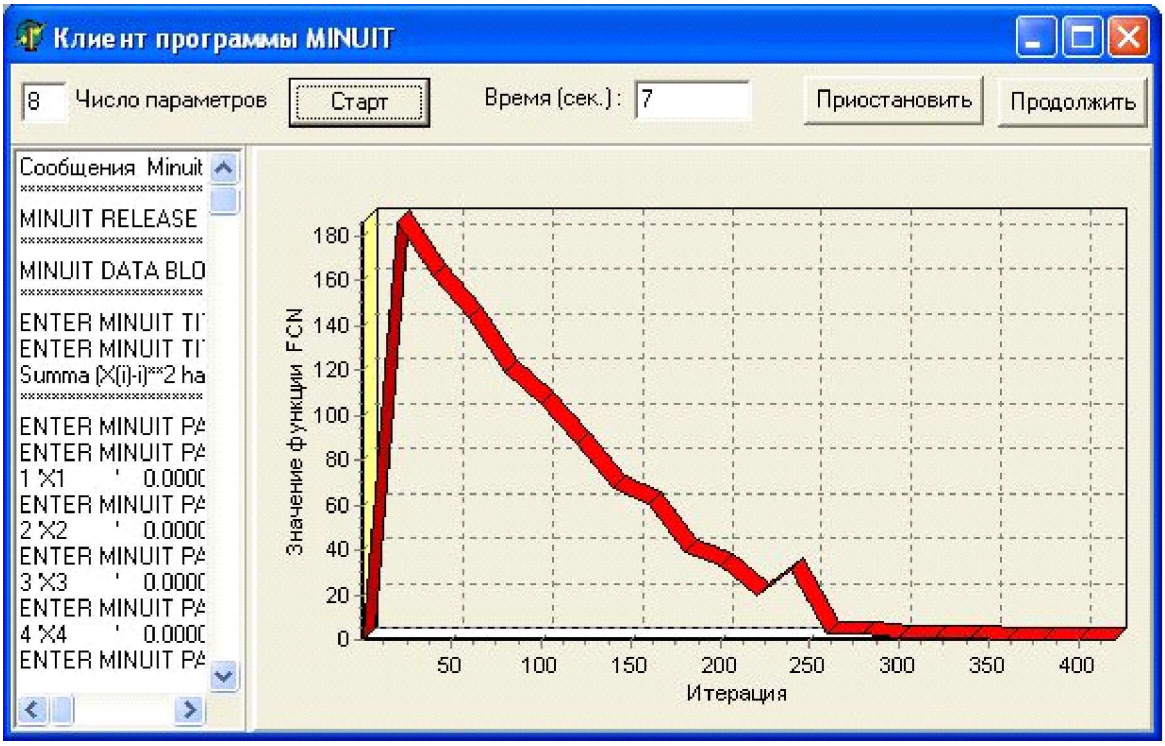

Рис. 2: Клиент программы MINUIT, выполняющий визуализацию процесса минимизации.

Таким образом, этот пример демонстрирует следующие достоинства F2F-технологии:

- интеграцию старых фортранных программ с современными графическими приложениями;
- простоту изготовления клиентских приложений на основе типового клиента;
- появление новых привлекательных свойств у вычислительных программ.

Материалы выложены на сайт библиотеки JINRLIB ЛИТ ОИЯИ:
www.jinr.ru/programs/jinrlib/f2f-technology/index.html

Список литературы

[1] S.I.Feldman, P.J.Weinberger, A Portable Fortran 77 Compiler. UNIX Time Sharing System Programmer's Manual, Tenth Edition, Volume 2, AT&T, Bell Laboratories, 1990.

[2] MPI: The complete Reference. MIT Press, Cambridge, Massachusetts. 1997.

[3] В.В.Воеводин, Вл.В.Воеводин. Параллельные вычисления. БХВ-Петербург, 2002.

[4] А.П.Сапожников, Т.Ф.Сапожникова. Реинжениринговая технология распределенных вычислений в локальной сети. Труды международной конференции "Распределенные вычисления и Грид-технологии в науке и образовании" (Дубна, 29 июня — 2 июля 2004 г.). 11–2004–205, Дубна, ОИЯИ, 2004. Стр. 183–190.

[5] CERN Program Library Long Writeup D506. James. F. MINUIT. CERN, Geneva, Switzerland.