го поиска диагностических проверок снижают количество задаваемых вопросов. Это приводит к эффективному использованию имеющихся ресурсов за счет сокращения трудозатрат медицинских работников, уменьшения объемов передаваемой информации по каналам передачи данных и нагрузки на имеющиеся приборы медицинской диагностики, обеспечивая при этом высокую объективность оценки и надежность работы системы диагностирования.

Литература

1. Гайдышев И. Анализ и обработка данных: спец. справочник. Спб: Питер, 2001.

2. Кобринский Б.А. Ретроспективный анализ медицинских экспертных систем // Новости искусственного интеллекта. 2005. № 2.

ВЫБОР ЯЗЫКА ВЫСОКОГО УРОВНЯ ДЛЯ РЕАЛИЗАЦИИ ВЫЧИСЛИТЕЛЬНОГО ПРИЛОЖЕНИЯ

Л.И. Воронова, д.ф.-м.н.; М.А. Григорьева

(Российский государственный гуманитарный университет, г. Москва, magsend@gmail.com)

В статье обосновывается выбор языка высокого уровня Fortran для реализации модулей, обеспечивающих высокопроизводительные вычисления в информационно-исследовательской системе «Шлаковые расплавы». Приведено описание метода интеграции вычислительных Fortran-приложений с оболочкой CORBA.

Ключевые слова: *Fortran, высокопроизводительные вычисления, Corba, интеграция, вычислительное приложение.*

Для эффективной разработки сложных вычислительных приложений важен выбор языка программирования. В статье обосновывается выбор языка реализации для подсистемы молекулярно-динамического моделирования новой версии *информационно-исследовательской системы* (ИИС) по результатам тестирования скорости вычислений на различных языках программирования высокого уровня.

В Российском государственном гуманитарном университете ведется разработка ИИС «Шлаковые расплавы». ИИС предназначена для компьютерного моделирования свойств многокомпонентных оксидных расплавов методом *молекулярной динамики* (МД) в режиме удаленного доступа и обеспечивает реализацию комплексных компьютерных экспериментов для моделей с большим числом (10^4–10^5) частиц [1].

На рисунке 1 представлена структура ИИС версии 9.0 (2009 г.).

Основной функционал распределяется в ИИС следующим образом:

- *web-клиент* реализует интерфейс пользователя с клиентской стороны;
- *web-сервер* обеспечивает удаленный доступ к другим компонентам ИИС; *Apache Cocoon* – надстройка над *web*-сервером *Apache Tomcat*, является средой для публикации динамического *web*-контента;
- *сервер БД* осуществляет хранение данных по проведенным экспериментам, а также справочной информации по физической химии;
- *сервер приложений* содержит вычислительные модули: *MD* – моделирование методом молекулярной динамики, *SGR* – статистико-геометрическое моделирование, *MNDO* (*Modified Neglect of Diatomic Differentional Overlap*) – полуэмпирический квантово-химический метод [1].

В рассматриваемой ИИС все вычислительные модули реализованы на языке *C++* с использованием *объектно-ориентированного программирования* (ООП) и распределения вычислений с помощью промежуточного ПО – технологии *CORBA*. Однако разработчики отмечают низкую производительность при расчетах [2], что делает необходимым пересмотр выбранного метода реализации, основываясь на опыте разработок научно-технических ИС подобного масштаба.

Решающим фактором при высокопроизводительных вычислениях является скорость вычислений, которая и определяет выбор языка программирования. Чтобы выбрать оптимальное решение, необходимо сравнить производительность вычислений на разных языках. Для тестирования были взяты три языка программирования – *Fortran*, *C++*, *Java*.

Fortran. В образовательной и научной среде бытует мнение о предпочтительности ООП на

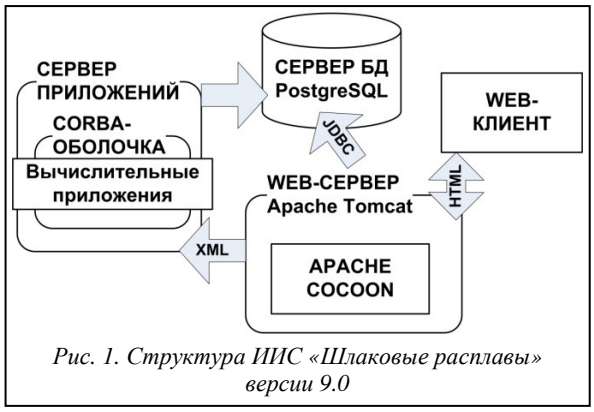

Рис. 1. Структура ИИС «Шлаковые расплавы» версии 9.0

языке *C++* для решения практически любой задачи. Однако сегодня существует достойная ему альтернатива – язык *Fortran*.

Fortran разрабатывался для решения сложных вычислительных задач и в настоящее время занимает лидирующее положение среди языков программирования, ориентированных на решение научно-технических задач. Особенно актуальным является применение *Fortran* с использованием современных параллельных вычислительных систем. Оптимизирующие компиляторы для *Fortran* были включены в список десяти наиболее выдающихся достижений *Computer Science* XX века.

C++. Универсальный обьектно-ориентированный язык программирования *C++* изначально предназначался для обеспечения удобства программирования. За исключением второстепенных деталей, *C++* является надмножеством языка программирования *C*. Помимо возможностей, которые дает *C*, *C++* предоставляет гибкие и эффективные средства определения новых типов – объектов. При правильном использовании этот метод позволяет сокращать программный код, создавать программы, легкие для понимания и контроля.

Огромный набор предоставляемых возможностей от побитового управления данными до форматированного ввода/вывода, а также относительная независимость и стабильность выполнения программ на разных машинах сделали язык *C++* чрезвычайно популярным в среде программистов.

Java разрабатывался для бытовой электроники, но впоследствии стал использоваться для написания клиентских приложений и серверного ПО. Программы на *Java* транслируются в байт-код, выполняемый виртуальной *Java*-машиной (*JVM*) – программой, обрабатывающей байтовый код и передающей инструкции оборудованию как интерпретатор, но с тем отличием, что байтовый код обрабатывается значительно быстрее текста.

Достоинство подобного способа выполнения программ в полной независимости байт-кода от операционной системы и оборудования, что позволяет выполнять *Java*-приложения на любом устройстве, поддерживающем виртуальную машину.

Тестирование выбранных языков на производительность при расчетах

Скорость тестировалась на компьютере *Inter(R) Core(TM) 2 Duo* с 2 Гб оперативной памяти. Операционная система *Windows XP SP3*. Среда проведения тестирования – *Eclipse GANYMEDE (JDK 6.0)*. Для тестирования программ на языке *C++* были использованы расширение *Eclipse CDT* и библиотека *Cygwin* – набор свободных программных инструментов, позволяющих превратить *Microsoft Windows* и *Windows NT* различных версий в подобие *UNIX*-системы. Для тестирования программ на языке *Fortran* использовано расширение *Eclipse – Photran*. Тестирование заключалось в реализации простого аглоритма, выполняющего около 300 миллионов вычислительных операций.

Пример кода на *Fortran*:

```
DO J=1,10000
    DO I=1,10000
        s3=SIN(I+J)
     END DO
END DO
```

Результаты тестирования: *Fortran* – 6 мс., *C++* – 11 мс., *Java* – 13 мс.

Полученные результаты показывают существенное превосходство языка *Fortran* по производительности при сложных вычислениях. Потому для реализации подсистемы моделирования выбор этого языка будет оправданным и оптимальным.

Разработчикам *Fortran* удалось найти компромисс между удобством программирования и эффективностью программ, написанных на этом языке. Синтаксис языка строится таким образом, чтобы обеспечить максимальную эффективность автоматической оптимизации исполняемого кода.

Выделим достоинства языка *Fortran* для научно-технических вычислений:

- изначальная ориентация на решение научно-технических задач (что отражено в названии *FORmula TRANslator* – транслятор формул);
- портируемость – высокая степень переносимости исходного кода между различными платформами;
- высокая эффективность исполняемого кода, что объясняется многолетней отработкой алгоритмов компилятора и использованием простых конструкций языка;
- огромный объем готовых математических наработок в виде коллекций библиотек.

В пользу языка *Fortran* можно привести сравнение функциональности операторов высокоуровневых языков с функциональностью языка *C* [3]: *C* – 1, *C++* – 2,5, *Fortran* – 2, *Java* – 2,5.

Одним из условий реализации вычислительных приложений в подсистеме МД-моделирования является необходимость использования распределенных вычислений для повышения производительности.

Интеграция Fortran с CORBA

CORBA (*Common Object Request Broker Architecture*) – стандарт промежуточного уровня, разработанный группой *OMG* и использующий *IDL* (*Interface Definition Language*) для определения интерфейса обмена данными между распределенными объектами. Так как *IDL* может использоваться с любым языком программирования, например, *C++*, *Java*, *Smalltalk*, существующие приложения могут быть интегрированы в новое приложение без переписывания программного кода.

В объектной модели *OMG* объектом является инкапсулированная сущность. Методы этого объекта могут быть доступны через интерфейсы, определенные с помощью *IDL*. Клиенты запрашивают объекты для выполнения методов (или сервисов), но реализация и местоположение каждого объекта скрыты от клиента. Коммуникация между клиентами и объектами обеспечивается *ORB* (*Object Request Broker*), ключевым компонентом архитектуры *CORBA*. При компиляции *IDL*-файла *ORB* генерирует программную заглушку (*stub*) и структуру программы (*skeleton*), через которые клиент может вызывать методы серверного объекта, находящегося или на том же компьютере, или на сетевом.

ORB ответственен за нахождение объекта, который может выполнить запрос, за передачу параметров, вызов метода, возвращение результатов клиенту.

При этом клиенту не важны местоположение объекта, язык реализации, операционная система и другие системные аспекты, не являющиеся частью объектного интерфейса.

CORBA является широко распространенным промежуточным стандартом для распределенных объектных вычислений. Он помогает разработчикам крупных распределенных приложений решать свои задачи в гетерогенном сетевом окружении, а также предоставляет стандартный интерфейс, обеспечивающий прозрачный обмен управляющей информацией между компьютером и коммуникационной сетью.

Многие научные приложения по аэродинамике и трехмерным моделям реализованы на языке *Fortran*. Оснащение кодов *Fortran* объектами *CORBA* помогает увеличить возможность повторного использования кода. К сожалению, не реализовано *CORBA-IDL-to-Fortran* соответствия и нет метода прямого генерирования *CORBA*-объектов из *Fortran* без применения написанных самостоятельно упаковщиков *C++*.

При интеграции приложения на языке *Fortran* в *CORBA*-приложение необходимо, насколько это возможно, сохранить код на *Fortran* немодифицированным.

Большинство программ на языке *Fortran* используют *COMMON*-блок для распределения большого количества переменных между различными функциями. Блоки *COMMON* схожи с глобальными переменными языка *C*. В программном окружении *CORBA* глобальные переменные не могут использоваться для передачи значений переменных между объектами. Наиболее оптимальным решением является выделение блока *COMMON* и конвертирование этого блока в структурный атрибут на языке *C++*. Через атрибуты каждый компонент может инициализировать переменные и возвращать результат вычислений клиенту.

В качестве инструмента преобразования кода на *Fortran* в код на *C/C++* был выбран конвертер *f2C* как свободно распространяемый и широко используемый при решении научных задач.

Конвертер *f2C* позволяет транслировать исходный текст программы на языке *Fortran 77* на язык *C/C++*, что дает возможность создавать переносимые программы на *C++*, использующие исходный текст на *Fortran*.

При решении задачи генерирования объектов *C* из *Fortran* для *CORBA* необходимо конвертировать только типы данных, описания переменных и заголовки функций. Однако код, конвертируемый программой *f2C*, не отвечает требованиям *IDL*, поэтому его необходимо будет изменять затем вручную [4].

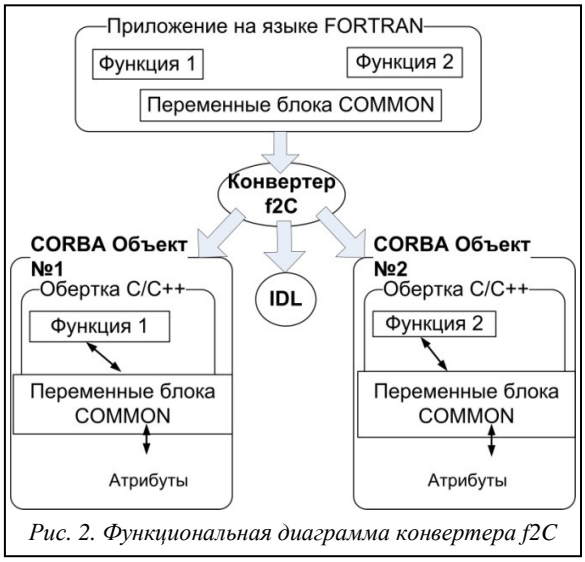

Рис. 2. Функциональная диаграмма конвертера f2C

На рисунке 2 представлена схема преобразования *Fortran* в *IDL* и в *CORBA*-объекты.

Из всех рассмотренных высокоуровневых языков программирования *Fortran* является наиболее производительным при реализации сложных математических вычислений. Учитывая возможность *Fortran* взаимодействовать с *CORBA* для организации распределенных вычислений, выбор данного языка является наиболее оправданным. Он позволит обеспечить максимальное быстродействие для подсистемы моделирования ИИС «Шлаковые расплавы».

Литература

1. Оптимизация информационного и программного обеспечения информационно-исследовательской системы «Шлаковые расплавы» / Л.И. Воронова [и др.]. РГГУ. М., 2009. 45 с.

2. Тетерин С.А., Воронов В.И., Воронова Л.И. Математическое моделирование металлургических шлаков МД-методом в ионно-ковалентной модели // Изв. Северо-Кавказ. региона. 2006. С. 16–21. (Технические науки. Прилож. № 10).

3. An Empirical Comparision of Seven Programming Languages (Prechelt, 2000).

4. Sang J., Kim C., Lopez I. Developing CORBA-based Distributed Scientific Applications From Legace Fortran Programs, NASA, 2000.