86

ОРДЕНА ЛЕНИНА
ИНСТИТУТ ПРИКЛАДНОЙ МАТЕМАТИКИ
АКАДЕМИИ НАУК СССР

В.Я. Карпов, Д.А. Корягин, А.А. Самарский.

ПРИНЦИПЫ РАЗРАБОТКИ ПАКЕТОВ ПРИКЛАДНЫХ ПРОГРАММ ДЛЯ ЗАДАЧ МАТЕМАТИЧЕСКОЙ ФИЗИКИ

Препринт № 86 за 1977г.

Москва.

А Н Н О Т А Ц И Я

Дан обзор проблем, стоящих перед разработчиками пакетов прикладных программ.

Ключевые слова: стандартная программе, пакет прикладных программ.

I. В настоящее время уровень развития вычислительной техники, программирования и численных методов для задач математической физики дает возможность поставить на повестку дня вопрос о создании математических моделей сложных физических и инженерно-технических процессов и объектов. Численное исследование таких моделей, достаточно полно отражающих основные свойства реального объекта, позволяет глубже понять природу объекта и уменьшить количество дорогостоящих физических экспериментов, необходимых обычно на различных этапах больших научно-технических программ.

Укажем в качестве примера две крупные проблемы комплексного характера, для решения которых необходимо предварительное всестороннее теоретическое исследование с использованием вычислительных машин.

Ключевая проблема ядерной энергетики состоит в создании ядерного реактора, являющегося сложным техническим сооружением. При его проектировании и управлении работой приходится совместно решать задачи из различных областей физики и техники. Это задачи ядерной физики и теории ядра, задачи переноса нейтронов и излучения, задачи теплопередачи, газовой динамики, теории упругости и прочности.

Другой крупнейшей научно-технической проблемой является проблема управляемого термоядерного синтеза. Она требует прежде всего изучения поведения плазмы при различных экстремальных условиях: при высоких температурах и плотностях, при низких плотностях, при наличии электромагнитных полей и термоядерных реакций. Процессы, протекающие в плазме, существенно нелинейны, и их теоретическое исследование представляет огромные трудности. Существует несколько проектов установок для получения термоядерного синтеза. Их сравнение между собой и оптимизация путем мате-

матического моделирования должны предшествовать строительству крупномасштабных дорогостоящих установок.

Сложный характер носят и такие важные проблемы как автоматизация обработки результатов физических экспериментов, решение задач химической кинетики, расчет строительных конструкций и другие.

2. Переход в машинных приложениях к решению проблем комплексного характера требует использования новых, более совершенных форм программирования и создания высокопроизводительных компонент программного обеспечения. С этим, естественно, связан пересмотр и уточнение некоторых сложившихся в программировании подходов и взглядов, а также определение основных текущих перспективных задач.

Начиная с середины 60-х годов в программировании можно довольно четко выделить два направления: прикладное программирование, которое охватывает вопросы создания алгоритмов и программ для решения задач в различных областях применения вычислительных машин, и системное программирование, имеющее своей целью разработку различных средств программного оборудования вычислительной машины. Деятельность коллективов, представляющих эти направления, характеризуется с одной стороны взаимным сотрудничеством, выражающемся прежде всего в том, что разработчики прикладных программ используют "средства производства" программ, которые создают программисты-системщики. В основе такого сотрудничества лежит общность главной цели, а именно, создание программного обеспечения, предоставляющего широкие возможности для эффективного решения задач на вычислительной машине. С другой стороны, работам, выполняемым в обоих направлениях, присущ определенный сепаратизм, вытекающий из различия проблематик направлений и отражающийся как в методах самого программирования, так и

в характере организации работы коллективов.

В течение ряда лет негативные аспекты этого сепаратизма не оказывали серьезного влияния на эффективность разработок прикладного программирования. Это объясняется тем, что одна из главных задач системного программирования - создание средств формального описания алгоритмов решения задач и аппарата перевода формализованных описаний на язык машины - была с самого начала поставлена в связи с необходимостью автоматизации программирования прикладных задач. Отметим, что постановки целого ряда других задач системного программирования также были почерпнуты из практики прикладного программирования. Эта практическая привязанность обусловила тот факт, что несмотря на указанный сепаратизм, средства, создаваемые системными программистами, широко использовались программистами-прикладниками, практически полностью удовлетворяя их системные запросы. В тех случаях, когда имеющихся системных средств оказывалось недостаточно, они разрабатывались прикладниками самостоятельно, причем, как правило, такие разработки сводились к модификации (в той или иной степени) наличных системных средств.

3. Подтверждение вышесказанному можно увидеть, рассматривая эволюцию программ решения задач математической физики. В период становления системного программирования степень сложности программы оценивалась обычно только с точки зрения ее объема (количества команд). В среднем это были программы, содержащие $10^3 \div 10^4$ команд. Для разработки таких программ вполне достаточно было использовать какой-либо проблемно-ориентированный язык (АЛГОЛ, ФОРТРАН) и соответствующий транслятор.

Затем с увеличением мощности вычислительных машин прикладники перешли к созданию больших программ (программных комплексов), каждая из которых предназначалась для решения сложной, но кон-

кретной задачи. При этом не только увеличился объем программ ($10^4 \div 10^5$ команд), но, что более важно, существенно усложнилась их логика. Этот этап эволюции прикладных программ характерен появлением нового подхода к разработке программ, получившего название модульного принципа программирования. В основе этого принципа лежит прежде всего возможность представить и решить полную задачу как совокупность относительно самостоятельных физических подзадач. Как правило, каждая физическая подзадача в свою очередь представляется совокупностью относительно самостоятельных математических задач. Использование такой структуры и было исходной идеей модульного программирования. Переход к модульному принципу программирования выразился в том, что решение задачи по существу обеспечивалось не одной программой, а группой взаимосвязанных программ, входящих в состав прикладного программного комплекса. Эти программы, специально оформленные так, чтобы обеспечить возможность их взаимодействия с различными другими программами, получили название модулей. В связи с новым подходом в языках программирования был развит аппарат подпрограмм (процедур), послуживший основой для реализации модульного принципа программирования, а также были разработаны специальные системные средства, обеспечивающие статическое или динамическое формирование комплекса определенной структуры.

В практическом отношении существенный недостаток таких программных комплексов заключался в том, что они ориентировались только на решение одной и реже нескольких конкретных задач определенного класса. Применение же подобного рода комплекса для решения родственной, но не предусмотренной при его создании задачи, несмотря на модульную структуру комплекса, как правило, было сопряжено с трудоемкими доделками, в конечном счете соизме-

римыми с работой по созданию полностью новой программы. Этот недостаток являлся следствием как несовершенства методов разработки прикладных программ, так и ограниченности возможностей, предоставляемых системными средствами.

Действительно, при разработке прикладных комплексов возможные варианты математических моделей учитывались путем включения в комплекс всех необходимых модулей и использования в процессе его работы так называемых управляющих параметров. Ясно, что при таком подходе гибкость и универсальность комплекса спредслялась тем, насколько предусмотрительно были выбраны модули и управляющие параметры. Следует также отметить, что с точки зрения каждой конкретной задачи такой программный комплекс характеризуется определенной избыточностью, поскольку он содержит "лишние", т.е. неиспользуемые при решении этой задачи модули и области данных. Что же касается ограниченности возможностей системных средств, то это выражалось, главным образом, не столько в отсутствии удобных средств для модификации текстов и перекомпановки структур прикладных программ (в том или ином виде такого рода услуги обеспечивались на развитых вычислительных системах), сколько в отсутствии унифицированных способов использования этих средств. Другими словами, решение вопросов согласованного использования и организации взаимодействия различных средств, например, таких как редактора текстов, транслятора - и системы управления заданиями, практически полностью возлагалось на прикладных программистов. Подчеркнем, что именно в связи с этим обстоятельством прикладники впервые серьезно столкнулись с необходимостью доработки, модификации и расширения возможностей имеющихся системных средств.

4. Мы уже отмечали, что настоящий этап развития программирования связан с переходом в машинных приложениях к решению проблем комплексного характера. В связи с этим мы хотим отметить три обстоятельства, которые на наш взгляд наиболее точно отражают текущее положение дел в программировании и взаимоотношения между двумя направлениями этой дисциплины:

а) проведение вычислительного эксперимента;

б) создание фондов программ;

в) универсальность системных средств.

Важнейшим моментом, определяющим характер работ в современных приложениях математической физики, является проведение вычислительного эксперимента. По существу, метод вычислительного эксперимента является основным теоретическим методом исследования сложных физических и инженерно-технических процессов и объектов. Для физических задач основные этапы вычислительного эксперимента состоят в следующем:

I. Выбор физического приближения и формулировка математической модели (как задачи математической физики).

2. Выбор дискретной модели, аппроксимирующей исходную математическую задачу (например, построение разностной схемы) и разработка вычислительного алгоритма для решения уравнений.

3. Создание программы для реализации вычислительного алгоритма.

4. Проведение расчетов и обработка полученной информации.

5. Анализ результатов, сравнение с физическим экспериментом, пересмотр и уточнение физической модели и, если нужно, повторение всех этапов сначала.

Указанные пункты I÷5 характеризуют один цикл вычислительного эксперимента. Если целью изучения некоторого физического процесса является его оптимизация (то есть выяснение и создание условий, при которых процесс протекает оптимально по каким-либо характерным параметрам), то проведение вычислительного экспери-

мента требует повторения указанного технологического цикла до тех пор, пока не будет получен искомый результат. С точки зрения программирования вычислительный эксперимент характерен тем, что для каждой физической модели с целью установления соответствия между физическим экспериментом и вычислительным экспериментом необходимо решать большое число вариантов (варьируя определяющие параметры задачи) и, кроме того, менять (уточнять) саму физическую модель. Эта особенность ("многовариантность" и "многомодельность") вычислительного эксперимента проявляется в многократных изменениях программы, реализующей вычислительный алгоритм, причем эти изменения касаются как структуры программы в целом, так и отдельных фрагментов программной реализации алгоритма.

Фактически, разработка программы для вычислительного эксперимента выливается в создание крупной программной системы (объемом порядка $10^5 \div 10^6$ команд), характеризующейся большим числом компонент и многообразием их взаимодействия. По-видимому, единственный реальный путь обеспечения вычислительного эксперимента программными средствами, позволяющими быстро собирать программы и моделировать вычислительный алгоритм и физическую модель, заключается в использовании модульного принципа программирования в сочетании с развитыми системными средствами. Плодотворность модульного принципа программирования обеспечивается тем важным обстоятельством, что описание одних и тех же физических процессов входит в различные комплексные проблемы. Кроме того, различные физические процессы часто описываются одними и теми же уравнениями. Например, одними и теми же уравнениями описываются такие процессы как диффузия, теплопроводность, намагничивание. Различие между процессами проявляется лишь в физическом

смысле коэффициентов уравнений и искомой функции. Следовательно, при оптимальном выборе множества математических и физических модулей, их число может оказаться меньше числа решаемых с помощью них физических задач.

В связи с этим уместно обратить внимание на второе обстоятельство, которое является итогом предшествующих усилий программистов и дает основания для оптимизма. К настоящему времени в прикладном программировании накоплен и возрастающими темпами пополняется значительный фонд программных средств, то есть описанных на языках программирования различных математических методов и алгоритмов решения отдельных задач. В ряде случаев эти программные средства хорошо проверены, документированы и широко доступны. В качестве примера можно привести отечественный Государственный фонд алгоритмов и программ [1] и два международных фонда физических программ: библиотеку при Королевском университете в Белфасте (Северная Ирландия) [2] и библиотеку реакторных программ в Испре (Италия) [3]. Наличие достаточно полного фонда программных средств позволяет перейти на качественно новый уровень в прикладном программировании. Теперь можно и нужно ставить на повестку дня вопрос о создании развитого программного обслуживания отдельных классов прикладных задач или, рассматривая вопрос шире, отдельных разделов прикладной деятельности. При этом обслуживание должно быть настолько полным, что разработка частной программы из данного класса, характеризующейся будь то большим числом используемых модулей, либо их сложностью и возможностью многократного изменения, либо многовариантностью их взаимодействия, либо совокупностью всех этих факторов, уже не являлась бы уникальной задачей, а представляла бы собой всего лишь один из рабочих моментов конкретной

прикладной деятельности.

Совершенно очевидно, что такая постановка вопроса предполагает использование адекватных системных средств, что мы уже отмечали, говоря о программном обеспечении вычислительного эксперимента. И теперь мы рассмотрим третье обстоятельство, касающееся возможностей существующих системных средств с точки зрения современного прикладного программирования. Дело в том, что разрабатывавшиеся до сих пор системные средства, несмотря на их практическую направленность, как правило, были ориентированы на некоего "обобщенного" пользователя. Это, конечно, обеспечивало их весьма широкую применимость, но в то же время приводило к тому, что с точки зрения каждой конкретной прикладной деятельности они оказывались не вполне адекватными. Иначе говоря, возможности, реализуемые системными средствами, определялись исходя из учета общих, а не специфических прикладных интересов. По мере расширения сфер прикладного программирования и совершенствования его методов, выяснилось, что услуг, обеспечиваемых универсальными системными средствами (например, средствами отладки) не достаточно для успешного проведения работ, а также, что более характерно, использование таких средств для специальных целей (в частности, редактирования, модификации и сборки программ) сопряжено с трудоемкими операциями. На наш взгляд это следствие сепаратизма, присущего разработкам прикладного и системного программирования, на современном этапе становится тормозом развития машинных приложений, связанных с решением комплексных проблем.

5. Проведенные рассмотрения позволяют сделать вывод о том, что одним из главных направлений работ в современном программировании должно стать создание проблемно-ориентированных програм-

мных систем, называемых пакетами прикладных программ. Причем успех на этом направлении может быть достигнут только путем объединения усилий коллективов прикладных и системных программистов.

Под пакетом прикладных программ мы понимаем комплекс взаимосвязанных прикладных и системных программ, обеспечивающих адекватное покрытие некоторой прикладной деятельности. Не претендуя на единственность определения пакета, а также учитывая наличие других определений, мы хотим несколько подробнее пояснить использованные здесь термины.

Прежде всего отметим, что всякая конкретная прикладная деятельность характеризуется двумя факторами. Во-первых, предметной областью, то-есть тем, на что направлен труд программиста, и представляющей собой совокупностью решаемыхим прикладных задач, и, во-вторнх, дисциплиной работы, т.е. совокупностью приемов, правил, принятых при разработке, отладке и эксплуатации программ. Адекватность покрытия означает достаточную полноту средств, связанных как с первым, так и со вторым факторами, определяющими конкретную прикладную деятельность. Другими словами достаточно полным должен быть как состав модулей, служащих материалом для сборки программ из данной предметной области, так и состав средств хранения, редактирования модулей, сборки программ и выполнения других видов принятых в данной прикладной деятельности работ. При этом в общем случае адекватность покрытия следует рассматривать во времени, то-есть всякое изменение в прикладной деятельности должно учитываться путем включения в пакет новых прикладных или системных программ. Таким образом, можно говорить о "динамической" адекватности покрытия.

Если интерпретировать конкретные компоненты программного обеспечения как показатели отдельных квалификаций, которыми обладает вычислительная машина, то нельзя не признать, что в настоя-

щее время уровень фактической прикладной квалификации большинства вычислительных машин ниже уровня их потенциальной прикладной квалификации, то есть не соответствует богатому фонду прикладных программ, о котором мы уже говорили. Дело в том, что к сожалению, имеющиеся прикладные программы, как правило, не дополнены соответствующими системными возможностями, что затрудняет их использование.

Из сказанного выше следует, что основной задачей пакетной проблематики в настоящее время надо считать повышения уровня прикладной квалификации вычислительных машин путем использования методов системного программирования на базе существующего и развивающегося фонда прикладных программ.

6. Организационно пакет прикладных программ можно представить состоящим из двух частей: функционального и системного наполнений.

Функциональное наполнение отражает специфику предметной области пакета и включает, например:

- совокупность модулей, используемых при составлении программ для решения задач данной предметной области;
- набор стандартных схем счета, определяющих модули, из которых состоит программа решения той или иной типичной задачи;
- набор описаний, отражающих различные функциональные и программно-эксплуатационные характеристики модулей и стандартных схем счета, входящих в пакет.

Требования на форму представления и организацию элементов функционального наполнения устанавливаются обычно при определении системных средств пакета.

Системное наполнение является административным органом пакета, отражающим дисциплину работы с пакетом. Системное наполнение может включать, например, такие компоненты:

- язык заданий, являющийся средством общения пользователя с пакетом;

- архив, являющийся системой хранения элементов функционального наполнения и олужебной информации пакета;

- монитор, представляющий собой совокупность программных средств, обеспечивающих операционные возможности пакета.

Следует отметить, что системное наполнение может быть в известной степени инвариантным относительно предметной области, то есть системное наполнение, разработанное для некоторой предметной области, может оказаться легко адаптируемым для других предметных областей.

7. Разработка пакета прикладных программ сопряжена с решением целого ряда проблем, из которых наиболее важными нам представляются следующие.

Проблемная ориентация. Первая проблема, решение которой предшествует собственно разработке пакета, состоит в определении его проблемной ориентации. По существу эта проблема сводится к решению двух вопросов: определению предметной области, т.е. класса задач, для которых предназначен данный пакет программ, и дисциплины работы с пакетом. Функциональное наполнение пакета должно разрабатываться с целью обеспечения достаточно полного материала для создания программ из этой предметной области, а системное наполнение должно обеспечивать принятые в ней виды работы. Совокупность конструктивных элементов и конструктивных средств пакета и обеспечивает отмеченное выше повышение квалификации вычислительной машины.

Архитектура пакета. Едва ли не самой главной проблемой, с которой приходится сталкиваться при разработке пакета, является выбор его архитектуры, то есть представляющегося пользователю внешнего вида пакета. Архитектура пакета отражается прех-

де всего во внешних спецификациях его возможностей и описании входного языка или языков. Успешное решение этой проблемы служит залогом эффективности функционирования и простоты эксплуатации пакета. По-видимому, непосредственно после определения архитектуры необходимо детально рассмотреть и установить основные принципы документирования пакета.

Модульный анализ. С целью выявления различных типов модулей, как конструктивных элементов на всех этапах функционирования пакета, должен быть проведен модульный анализ алгоритмов, используемых в данной предметной области, а также проведены исследования общей структуры пакета. Здесь мы считаем необходимым подчеркнуть, что определение понятия "модуль", по-видимому, может быть дано только применительно к конкретному пакету. Это понятие фактически отражает принятый в пакете способ сборки программ и поэтому вряд ли следует придерживаться традиционного понимания под модулем некоторой функционально законченной программной единицы. Эффективная реализация принципов построения программ из модулей сопряжена с развитием методов и средств формального описания семантики и синтаксиса модулей. Следует заметить, что практически приемлемых решений в этом направлении до сих пор получено очень мало.

Операционное обеспечение. Разработчики пакета должны определить набор системных средств , обеспечивающих его эффективное функционирование. После этого необходимо провести исследование способов реализации этих средств на базе штатного операционного обеспечения вычислительной машины. Весьма существенно, чтобы при решении этих вопросов не были оставлены без внимания такие показатели как простота эксплуатации и внедрения пакета. Практика показывает, что достаточно часто степень распространенности той или иной системной разработки определяется главным образом этими показателями.

Интерактивное взаимодействие. Характерной чертой современных развитых программных систем является возможность интерактивного взаимодействия с ними, или, как еще принято говорить, возможность эксплуатировать их в режиме диалога. Такой режим общения с пакетом представляется весьма перспективным, особенно если учесть, что в некоторых прикладных задачах режим диалога является единственно практически приемлемым способом управления процессом обработки данных. В этой связи необходимо провести изучение общих принципов взаимодействия пользователя с пакетом прикладных программ, исследовать специфику решения задач в режиме диалога и определить ее влияние на структуру пакета. Полученные при этом результаты могут быть использованы при разработке конкретных языков общения с пакетом и системных средств организации диалога.

Информационное обеспечение. Даже неполное перечисление элементов функционального наполнения пакета, сделанное выше, позволяет судить о том, что практически невозможно достигнуть высокого уровня организации работы над такой обширной базой данных без соответствующих средств информационного обеспечения. В круг вопросов, связанных с созданием информационного обеспечения, входит исследование средств описания предметных областей пакетов, описаний модулей, в том числе для справочных целей, и описания вычислительной среды, в которой осуществляется эксплуатация пакета. Учитывая, что использование службы информации всегда сопряжено с определенными затратами времени и памяти, необходимо особое внимание уделить вопросам оптимизации информационного обеспечения. Не конкретизируя возможные подходы к реализации информационной службы, мы хотим тем не менее указать, что использование в ней только средств простейшего информационного поиска, по-видимому, окажется не достаточным для обслуживания такой об-

ширной и динамически изменяемой базы данных, какой является функциональное наполнение пакета.

Автоматическое планирование вычислений. Мы уже отмечали, что основной задачей пакетной проблематики является повышение уровня прикладной квалификации вычислительной машины. В непосредственной связи с такой постановкой находятся исследования и разработки средств автоматического построения программы на основе математической модели решаемой задачи, описаний семантики модулей функционального наполнения и описаний физических соотношений, существующих в данной предметной области. На наш взгляд не следует стремиться к реализации полностью автоматического способа сборки программы. Более реальным и практичным представляется компромиссное решение, при котором задание пакету формулируется как с использованием средств процедурного описания (то есть явно определяющих все данные и выполняемые над ними преобразования), так и средств непроцедурного описания (то есть указывающих только исходные данные и конечную цель некоторого фрагмента процесса). При таком подходе пользователь при желании может точно указать машине "что и как" надо делать. Вместе с тем он может предоставить ей определенную самостоятельность в выборе способа решения отдельных подзадач, то есть поручить пакету автоматически построить отдельные части программы.

8. Заканчивая обсуждение отдельных аспектов пакетной проблематики, мы хотим подчеркнуть, что принципиальным моментом в организации работ над пакетом является обязательность участия в решении каждой частной задачи (во всяком случае на этапе постановки задачи и составления технического задания на программу соответствующей компоненты пакета) как прикладных, так и системных программистов. Только при таком подходе может быть достигнуто органичное взаимодействие функционального наполнения с системными средствами и обеспечены высокие эксплуатационные характеристики

пакета.

В заключение хочется высказать следующее соображение. В настоящее время пакетная проблематика переживает начальную стадию своего развития. В целом ряде уже имеющихся программных разработок, которые можно отнести к данной проблематике, наблюдается отсутствие единой терминологии, различие в подходах и понимании задач, стоящих при создании пакета прикладных программ. Поэтому представляется чрезвычайно актуальной потребность в четком выделении круга вопросов, составляющих пакетную проблематику и в установлении единой терминологии. Мы надеемся, что данная работа позволит сделать определенный шаг в этом направлении.

ЛИТЕРАТУРА

1. АЛГОРИТМЫ И ПРОГРАММЫ. ГОСУДАРСТВЕННАЯ ПУБЛИЧНАЯ НАУЧНО-ТЕХНИЧЕСКАЯ БИБЛИОТЕКА СССР.
2. 'THE CPC PROGRAM LIBRARY', COMPUTER PHYSICS COMMUNICATIONS, V. 1, 473-476 (1970).
3. ENEA COMPUTER PROGRAMME LIBRARY, ISPRA(VARESE), ITALIA.